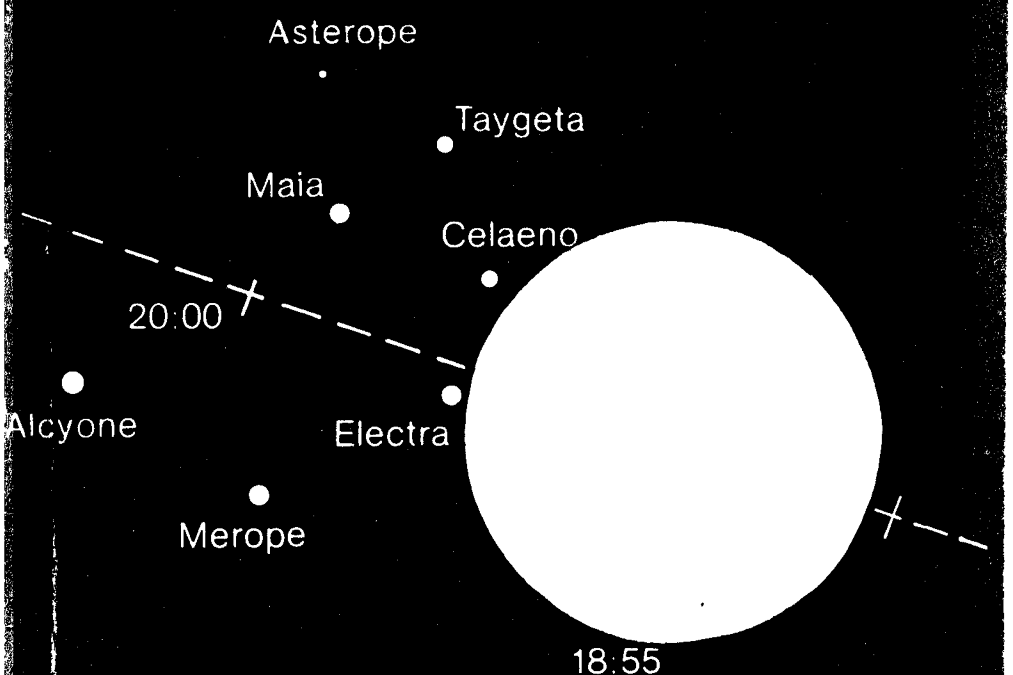
О. Монтенбрук, Т. Пфлегер

Астрономия с персональным компьютером



Издательство «Мир»

O. MONTENBRUCK, T. PFLEGER

ASTRONOMIE
MIT DEM
PERSONAL COMPUTER

Mit 37 Abbildungen

Springer-Verlag Berlin Heidelberg New York
London Paris Tokyo Hong Kong

О.Монтенбрук, Т.Пфлегер

Астрономия с персональным компьютером

Перевод с немецкого
канд. физ.-мат. наук А.Г. Тоточавы
под редакцией
канд. физ.-мат. наук Н.Н. Самуся

ББК 39.6
А91
УДК52+529.71/.74

Астрономия с персональным компьютером: Пер. с нем./Под ред.
А91 О. Монтенбрука, Т. Пфлегера. —М.: Мир, 1993. —279 с., ил.
ISBN 5-03-002452-2

Книга автора из ФРГ содержит большое число программ, обеспечивающих вычисление юлианских дат, преобразование координат, расчеты моментов восхода и захода светил, положение планет, комет и Луны, вычисление орбит по позиционным наблюдениям, обработку астрометрических измерений по фотографиям. Все программы составлены на языке Паскаль, они сопровождаются пояснением алгоритмов и вычислительных методов. Может служить учебным пособием.

Для астрономов-любителей и профессионалов.

А $\frac{1605010000-026}{041(01)-93}$ 50-92 ББК 39.6

Редакция литературы по физике и астрономии

ISBN 5-03-002452-2 (рус.)
ISBN 3-540-51386-8 (нем)

ПРЕДИСЛОВИЕ РЕДАКТОРА ПЕРЕВОДА

Помнится, еще в школьные годы, мне посчастливилось попасть на экскурсию в машинный зал ЭВМ БЭСМ-2. Тогда уже существовали и более совершенные машины, но учитель поступил правильно: именно от такой экскурсии впечатление осталось на всю жизнь. В огромном помещении стоял ламповый мастодонт, было жарко, в динамиках что-то трещало и пищало, а мы, пробираясь между шкафами с аппаратурой, думали о грандиозности достижений человеческого разума.

Достижение это, действительно, было большим (не по размерам, а по значению) для своего времени. Задачи, для решения которых прежде несколько человек должны были целый месяц крутить ручки арифмометров, теперь решались за считанные часы или даже минуты. Стало возможным взяться за решение задач, до той поры вовсе недоступных.

Большая машина требовала немалого коллектива для технического обслуживания. Машин было мало, а пользователей—много, поэтому задачи «отлеживались» в очереди на счет Вначале пользователям приходилось программировать в машинных кодах, т.е не машина приспосабливалась к человеку, а человек к машине. Однако вскоре появились первые языки программирования более высокого уровня, например Алгол-60, сильно облегчавшие работу программиста.

Современная персональная ЭВМ стоит на рабочем столе, не занимая его целиком, но намного превосходит незабвенную БЭСМ-2 и по оперативной памяти, и по объему внешних запоминающих устройств, и по быстродействию. Правда, в нашей стране слово «персональная» в названии таких ЭВМ—чаще всего просто перевод с «западного»; обычно такая машина эксплуатируется группой пользователей. Тем не менее и у нас появляется все больше людей, у которых есть ПЭВМ дома, а на западе это норма для семьи научного работника и не редкость для других семей. После первой волны увлечения компьютерными играми хозяин ЭВМ начинает задумываться над более серьезными применениями. И если научный работник сразу думает о своих повседневных вычислительных потребностях, то человек другой специальности, интересующийся наукой, привлекает ЭВМ к своему хобби.

Среди друзей науки немало любителей астрономии. Те из них, кто действительно серьезно изучает астрономию, могут своими наблюдениями принести ей реальную пользу—этим, пожалуй, астрономия принципиально отличается от других наук, где возможности непрофессионала крайне ограничены Но для серьезных наблюдений надо знать, какие явления происходят на небе и в какое время Астрономические календари и ежегодники дают нужную информацию не всегда достаточно подробно, да и не всегда оказываются под рукой в нужный момент. Так возникает интерес к самостоятельным расчетам астрономических явлений Некоторые виды расчетов возможны даже с калькулятором. Сводку полезных и очень точных формул можно найти, например, в книге Я. Меёуса «Астрономические формулы для калькуляторов» (М., «Мир», 1988 г.). Намного бóльшие возможности открывает ПЭВМ.

Чтобы предоставить пользователю не только формулы, но и программы, необходимо, чтобы используемый язык программирования был понятен его машине. Даже среди достаточно распространенных языков не все хорошо удовлетворяют этому требованию. Так, Бейсик имеет множество модификаций, и если опубликовать программы на Бейсике, то могут потребоваться немалые усилия для перевода программ на тот вариант Бейсика, которым располагает пользователь. Среди современных языков программирования наиболее независимым от конкретной реализации, пожалуй, является Паскаль. Этот язык возник как дальнейшее развитие упоминавшегося в начале предисловия Алгола и сохранил важную особенность Алгола, которая не всем нравилась, но оказалась очень полезной: в Паскале строго определяются все его конструкции; Паскаль требует строгого описания каждой используемой в программе переменной. Это та плата, которая неизбежна за преимущество универсальной применимости Паскаль-программ.

Предлагаемая вашему вниманию книга О. Монтенбрука и Т. Пфлегера «Астрономия с персональным компьютером» была опубликована в Германии в 1989 г. и сразу же переведена на английский язык. Она содержит программы на языке Паскаль, позволяющие самостоятельно производить расчеты многих астрономических явлений: восходов и заходов светил, солнечных затмений, положений планет, комет. Есть программы для пересчета астрономических координат, вычисления орбит, определения координат светила по астрометрическим измерениям на фотоснимках. Программы построены на принципах структурного программирования, практически каждая подпроблема решена в виде подпрограммы, которая может найти применение и вне основной программы.

Конечно, нельзя рассчитывать даже с помощью такого прекрасного справочника «без труда выловить рыбку из пруда». Среди пользователей ЭВМ существует полушутливое поверье, что в любой публикуемой программе всегда нарочно сделаны две ошибки, чтобы ею мог воспользоваться только тот, кто действительно этого заслуживает. А если серьезно, конечно, нужно разобраться и в программах, и в особенностях своего варианта паскалевского компилятора (несмотря на всю универсальность, о которой сказано выше). Советы о том, как преодолеть возможные трудности, тоже содержатся в книге. Думаю, она будет небесполезна и тем, у кого на ПЭВМ нет компилятора с Паскаля и нет желания переходить на Паскаль с привычного другого языка программирования: хотя затраты труда будут намного больше, все же, вероятно, удастся приспособить приведенные в книге алгоритмы. А войдя во вкус, вы сможете составить новые программы, решающие важные астрономические задачи, не представленные в книге. В качестве первого шага могу посоветовать оформить в виде отдельной программы (со своим вводом данных) реализацию метода наименьших квадратов и подумать над задачами поиска периодичностей, содержащихся в наблюдательных данных—скажем, в результатах фотометрии переменной звезды.

Хочу пожелать любителям астрономии и вычислительной математики успешных целенаправленных астрономических наблюдений с использованием расчетов по программам, которые вы найдете в книге.

Сентябрь 1991 г. *Н. Н. Самусь*

ПРЕДИСЛОВИЕ

Все, кто сегодня в качестве любителя или профессионала имеют дело с вычислениями астрономических явлений, неизбежно обращаются к компьютеру. Это стало особенно очевидным с тех пор, как персональный компьютер в роли вездесущего помощника занял постоянное место в нашей жизни. Широкому кругу потребителей прямо на письменном столе предоставлены сейчас такие возможности счета, которые еще недавно трудно было себе представить.

В той же мере, в какой улучшаются технические характеристики компьютера, увеличивается потребность в производительных, т. е. быстрых и точных программах. Возникает естественное желание по мере возможности отказаться от астрономических ежегодников.

Исходя из этих соображений, мы охотно поддержали предложение издательства написать книгу, содержащую компьютерные программы для решения основных задач сферической астрономии, небесной механики и вычисления эфемерид.

«Астрономия с персональным компьютером» предлагает читателю, самому пишущему программы, обширную библиотеку подпрограмм на языке Паскаль с готовыми решениями многих часто встречающихся задач. В их числе подпрограммы для всех распространенных преобразований координат, для вычислений времени и календарных дат, а также для решения задачи двух тел. Отдельные подпрограммы позволяют вычислять положение Солнца, Луны и планет с учетом взаимных возмущений. Благодаря широкому распространению языка Паскаль и отказу от специфических для разных машин команд, программы могут быть использованы как на всех распространенных компьютерах типа PC, так и на больших вычислительных машинах Представленные программы будут содействовать тому, чтобы пользователь мог сосредоточиться на интересующих его задачах, а не «изобретал колесо»

Каждая глава книги посвящена отдельной теме и заканчивается полной основной программой. Изложение начинается с простых вопросов, таких, как вычисление восходов и заходов светил, эфемерид Луны и планет, а затем рассматриваются более сложные, например затмения Солнца и покрытия звезд. Программы для астрометрической обработки снимков звездных полей и программы вычисления орбит позволяют определить элементы орбиты комет и астероидов. Ими смогут сразу же воспользоваться и читатели, не имеющие опыта программирования.

Сведения из астрономии и программирования, необходимые для решения каждой проблемы, изложены настолько подробно, насколько это необходимо для понимания представленной программы. Эти сведения дают возможность приспособить все программы к индивидуальным потребностям. Кроме того, тесное соединение теории и практики позволяет хотя бы частично упростить сложные зависимости по сравнению с тем, что предлагают классические учебники. Мы надеемся, что читатели этой книги получат серьезное подспорье для применения компьютера в астрономии.

Благодарим доктора Даниеля и доктора Михелса из издательства за приятную совместную работу и их интерес к изданию книги. Благодарим также всех наших друзей и коллег, которые своими идеями, инициативой и помощью при проверке текста и тестировании программ в значительной мере способствовали успеху нашей работы.

Август 1989 г. *О. Монтенбрук и Т. Пфлегер*

1. ВВЕДЕНИЕ

В последние годы наряду со снижением цен на малые компьютеры происходит непрерывное увеличение их мощности. Поэтому в настоящее время уже многие, интересующиеся астрономией, имеют в своем распоряжении персональные компьютеры и при этом естественно желают использовать их для астрономических вычислений. Какими же конкретными преимуществами наделяет компьютер своего владельца, если необходимые для наблюдений данные с успехом можно найти в многочисленных ежегодниках?

1.1. Примеры применения компьютеров

Рассмотрим прежде всего вычисление моментов восходов и заходов Солнца и Луны. С изменением места наблюдения заметно меняются и моменты восходов и заходов, чем пренебрегать нельзя. В ежегоднике эти моменты приведены, как правило, лишь для немногих мест на Земле. Часто при определении моментов восходов и заходов светил для любого пункта на Земле ограничиваются интерполяцией или применением номограммы. Будет значительно удобнее и точнее, если вычислитель без дополнительных усилий прямо сможет получить искомые данные.

Большую часть любого ежегодника составляют многостраничные таблицы положений Солнца, Луны и планет. Из данных этих таблиц зачастую только малая часть представляет для пользователя какой-то интерес. С помощью соответствующих программ можно вычислить действительно нужные величины и одновременно представить их в желаемой системе координат.

Еще важнее возможность самому вычислить орбиту небесного тела, например кометы. Для недавно открытых комет наблюдатель может вообще не иметь эфемерид или получить их слишком поздно. Но при этом достаточно иметь единственный набор элементов орбиты, чтобы довольно точно следить за движением кометы.

При наблюдениях покрытий звезд Луной требуется знать момент контакта для данного места наблюдения. Если в ежегоднике вообще есть такие предсказания, они ограничиваются, как правило, несколькими большими городами. В случае наблюдений, ведущихся в другом месте, приходится довольствоваться только оценками. Представленная в этой книге программа OCCULT позволяет осуществлять поиск покрытий и вычислять их моменты для любого числа звезд.

Многие любители астрономии занимаются астрофотографией. На самостоятельно полученном снимке с помощью опорных звезд, координаты которых известны, можно определить положение астероида или кометы. Для определения орбиты нужно иметь по крайней мере три пластинки, полученные в различные моменты времени. С их помощью можно определить элементы орбиты. Для этих действительно трудоемких вычислений мы приводим программы FOTO, ORBDET и COMET.

1.2. Астрономия и вычислительная математика

Знакомство с методами астрономических вычислений на компьютере было бы неполным без обсуждения достижимой точности. При этом прежде всего нужно заметить, что качество программ зависит от математического и физического описания соответствующей задачи и использованного алгоритма. Предположение о том, что использование вычислений с двойной разрядностью само по себе гарантирует правильность результата, уже много раз оказывалось ложным. Правильно работающая программа может дать неточный результат, если метод вычисления не обеспечивает лучшей точности.

Примером тому служит вычисление эфемерид в задаче двух тел. Здесь орбита планеты довольно просто представляется в виде эллипса. Наряду с силой притяжения Солнца, определяющей такую форму орбиты, действуют и другие силы со стороны остальных планет. Они вызывают периодические и вековые возмущения, которые для внутренних планет составляют около $1'$, а для внешних планет могут достигать даже 1°. Теория кеплеровых орбит представляет собой лишь приближенное описание истинных соотношений. Даже использование высокоточных вычислительных методов в этом случае не дает желаемых результатов. Поэтому для достижения большей точности обязательно нужно пользоваться сложным описанием планетных орбит, учитывающим взаимные возмущения планет. Одним применением в расчетах большего числа знаков мы немногого добились бы!

При разработке наших программ мы стремились к такой же точности, как у астрономических ежегодников. При вычислении координат Солнца, Луны и планет с помощью основных программ ошибки составляют примерно $1-3''$. Этой точности достаточно, например, для вычисления затмений Солнца или покрытий звезд и потому должно быть достаточно и для большинства других задач.

Для достижения высокой точности в первую очередь надо разобраться в определении применяемой системы координат. Практика показывает, что часто сравниваются данные, представленные в различных системах координат. И тогда, получив расхождение с данными ежегодника, делают вывод, что программа работает с невысокой точностью. Однако нередко обнаруженное расхождение объясняется всего лишь небольшим различием систем координат, которое пользователь не учел, заподозрив ошибку. Поэтому во всех соответствующих местах мы указываем важнейшие поправки, такие, как прецессия и нутация или же аберрация и время распространения света. Мы многократно подчеркиваем также различие всемирного и эфемеридного времени, которое почти всегда создает трудности. Частота подобных ссылок не должна смущать читателя. Мы даже надеемся, что такая форма представления и соответствующие изменения в программах способствуют лучшему пониманию практического значения и величины поправок. Иногда нам кажется необходимым повторять некоторые положения, чтобы каждая глава, представляющая четко ограниченную тему, могла быть прочитана вне зависимости от других глав. Но так как программы часто построены как продолжение одна другой, то в некоторых случаях все же необходимо последовательно прочитать несколько глав.

Эффективность программы зависит не только от математического описания задачи, но и от используемого метода счета. Эти аспекты рассматриваются по мере надобности в соответствии с конкретным приложением. Примером может служить использование угловых функций при вычислении периодических возмущений орбит Луны и планет. Применение теоремы сложения и рекурсивных соотношений для тригонометрических функций позволяет резко сократить время счета программ PLANPOS и LUNA. Если для решения определенной задачи нужно иметь много значений положения небесного тела на малых расстояниях друг от друга, можно использовать аппроксимацию полиномом Чебышева. При этом на ограниченном интервале времени можно получить общее выражение, позволяющее очень быстро и просто вычислить координаты без потери точности. Эта методика применялась в программах ECLIPSE и OCCULT. Другие аспекты вычислительной математики рассматриваются в связи с решением уравнений с применением метода Ньютона, правила ложного положения (метода секущих) и квадратичной интерполяции. Кроме того, для астрометрической обработки фотопластинок мы приводим простой, но надежный алгоритм решения уравнений.

1.3. Языки и техника программирования

При разработке наших программ мы старались соблюдать некоторые правила, которых рекомендуем придерживаться и нашим читателям. Эти правила характеризуются удачным названием «структурное программирование». В данном разделе мы хотим немного подробнее объяснить, какой смысл вкладывается в это выражение.

Прежде всего программы должны быть наглядными и читаемыми. Комментарии к важнейшим местам в тексте программы облегчают понимание программы не только потребителю, но и самому автору программы. Даже автор программы через некоторое время нередко забывает детали соотношений, и тогда он может восстановить их по своей документации, если позднее захочет вернуться к программе. По той же причине мы рекомендуем использовать выразительные имена переменных. Часто при этом английское обозначение бывает точнее. Поэтому в большинстве случаев мы ориентировались на английские названия. Применение блоков и пустых строк позволяет лучше понять логическую структуру. В исходном тексте место не экономят, так как компилятор игнорирует пустые строки.

Термин «структурное программирование» означает, что программа имеет возможно более ясное логическое построение. Современные языки программирования достигают этой цели, например, через структуры управления (петли, условные директивы) и через структурированные типы данных (records) или построением отдельных блоков программ в виде функций или подпрограмм. Насколько нужны или целесообразны безусловные переходы (команда GOTO), каждый читатель может решить сам. Мы довольно легко справляемся с возникающими трудностями без таких переходов и поэтому полностью от них отказываемся.

Для разработки программы в этом случае нужно пользоваться определенным методом, называемым нисходящей (top-down) стратегией. Это означает, что проблему прежде всего подробно анализируют и постепенно разбивают на малые задачи, и это продолжается до тех пор, пока программирование существенно не упростится. Такой подход кажется наиболее благоразумным и осмысленным. К сожалению, нередко программист действует как вычислитель и сообразно с этим формулирует свою программу. При таком образе действий получаются трудно обозреваемые программы, которые нелегко понять. Дальнейшие изменения в них обычно требуют больших усилий и «трюков», которые иногда может понять только автор программы. Этот способ программирования в лучшем случае подходит лишь для создания небольших несложных программ, которые либо мало применяются или вообще не понадобятся в дальнейшем. Поэтому составления таких программ желательно избегать.

Подпрограммы, решающие определенные задачи, должны быть по возможности составлены столь универсально, чтобы в дальнейшем их можно было использовать в других программах. Как раз при разработке астрономических программ почти всегда найдутся подпрограммы, которые можно использовать в других случаях. Примером тому служат подпрограммы для вычисления математических функций, пересчета координат из одной системы в другую или определения точных положений планет, Солнца и Луны. Было бы довольно обременительно каждый раз заново заниматься этими задачами. Поэтому необходимо иметь в своем распоряжении библиотеку функций в виде работоспособных и надежных подпрограмм, которая позволит сосредоточиться на возникающих проблемах, чтобы быстрее продвигаться к намеченной цели.

Функции и подпрограммы, приведенные в этой книге, представляют базовый блок такой астрономической библиотеки программ. В основном это:

* научно-технические функции и вспомогательные функции вычислительной математики,
* программы для вычисления времени и календарных дат,
* различные подпрограммы для решения задач сферической астрономии,
* специальные программы для учета прецессии и нутации,
* подпрограммы для вычисления эллиптических, параболических и гиперболических кеплеровых орбит, а также
* программы для вычисления точных положений Луны и планет с учетом различных возмущений.

Полный список отдельных подпрограмм представлен в приложении.

Для создания наших программ мы обратились к языку Паскаль. Основанием для этого послужило то обстоятельство, что Паскаль широко известен и применяется во всех распространенных системах компьютеров, включая и большие машины Все персональные компьютеры типа IBM и совместимые с ними могут использовать быстрый и недорогой компилятор TURBO PASCAL Борланда. TURBO PASCAL может быть приобретен и для компьютеров типа Apple фирмы Macintosh, и для операционных систем CP/M, работающих на старых 8-битовых машинах. Для компьютеров типа ATARI ST, особенно популярных в Германии, существует ST PASCAL PLUS фирмы CCD. На больших вычислительных машинах практически всегда имеется компилятор Паскаля.

В отличие от языка BASIC, который имеет различные диалекты в зависимости от машины, программы на Паскале могут быть почти полностью перенесены с одной машины на другую. Если программа составлена только с помощью стандартных элементов языка, то ею может пользоваться любой потребитель. При этом, возможно, потребуются незначительные изменения, которые могут быть внесены без особых усилий. Соответствующие указания по адаптации программ к конкретным компьютерам даны в приложении Возможность использовать программы для различных компьютеров сопряжена и с некоторыми ограничениями. Так, в представленных программах не используется графика или специальные возможности различных операционных систем. Читатель, желающий использовать сервисные возможности своей машины или вывести результат в виде графика, всегда сможет переделать программу по своему усмотрению.

Паскаль создан как язык для обучения и требует при программировании определенной дисциплины. На «обстоятельность» языка жалуются не только начинающие программисты, но она обеспечивает и определенные преимущества В частности, необходимость согласования всех переменных позволяет избежать многих ошибок. В языках FORTRAN и BASIC можно быстро вводить программу с клавиатуры, но каждое ошибочное нажатие клавиши будет вносить искажения в текст программы. Исправление этих ошибок обычно превращается в жестокое испытание. Паскаль является описательным языком, поэтому он выявляет ошибки уже при компиляции программы и предостерегает пользователя соответствующими указаниями. Проверка различных параметров одной функции или подпрограммы приобретает при этом особое значение. Только в том случае, когда все типы переменных описаны правильно, программа может быть оттранслирована без ошибок. Забыть при этом отдельные параметры невозможно. Такой компиляторный контроль значительно упрощает использование готовой библиотеки программ.

К сожалению, за это преимущество Паскаля приходится платить тем, что программа, написанная на этом языке, всегда должна транслироваться полностью. Более развитые языки, такие, как MODULA и ADA, при обращении к нужной информации способны выделять фрагменты, которые отличаются от подпрограмм, что позволяет транслировать отдельные модули при сохранении контроля. Это качество особенно важно при работе с большим пакетом программ. Читатели, пользующиеся довольно распространенным языком MODULA, не будут испытывать особенных затруднений при адаптации наших программ, так как PASCAL и MODULA очень похожи Новая версия TURBO PASCAL также обладает такими возможностями (units), которые, правда, выходят за рамки собственно Паскаля.

2. СИСТЕМЫ КООРДИНАТ

Для определения положения звезд и планет в астрономии используется несколько систем координат. При этом гелиоцентрические координаты, в которых за точку отсчета принимается Солнце, отличаются от геоцентрических координат, отсчитываемых от центра Земли. Эклиптические координаты характеризуют положение точки относительно плоскости орбиты Земли, в то время как экваториальные координаты измеряются относительно положения земного или небесного экватора. Кроме того, медленное перемещение этих поверхностей отсчета, вызванное прецессией, делает необходимым различать используемые системы координат по их равноденствию. Разумеется, в принципе можно было бы во всех случаях использовать единственную четко определенную систему координат. Их многообразие объясняется тем, что они имеют разные свойства, позволяющие выбрать систему, наиболее удобную для решения конкретной задачи.

Многообразие координатных систем часто приводит к необходимости решения довольно неприятной задачи — перевода координат из одной системы в другую. В этом случае может пригодиться программа КОКО (преобразование координат) С ее помощью можно делать взаимный пересчет эклиптических координат в экваториальные и геоцентрических в гелиоцентрические с учетом различия равноденствий. Преобразования, описанные ниже, будут представлены в виде отдельных подпрограмм. Они могут быть использованы во многих случаях и потому служат важнейшей основой последующих программ.

2.1. Начинать всегда нелегко

При программировании математических формул на языке Паскаль быстро обнаруживается, что не хватает некоторых средств, так как Паскаль в этом смысле беднее, чем, например, FORTRAN или BASIC. Однако эта проблема может быть успешно решена, если некоторые небольшие подпрограммы и функции подготовить самим

К их числу принадлежат тригонометрические и обратные им функции, которых либо вообще нет в языке Паскаль, либо они вычисляются в радианной мере

```
(*-----------------------------------------------------------*)
(* SN  Синус-функция (градусы)                               *)
(*-----------------------------------------------------------*)
FUNCTION SN(X:REAL).REAL,
  CONST RAD=0.0174532925199433,
  BEGIN
```

```
  SN:=SIN(X*RAD)
  END;
(*-----------------------------------------------------------*)
(* CS: Косинус-функция (градусы)                             *)
(*-----------------------------------------------------------*)
FUNCTION CS(X:REAL):REAL;
  CONST RAD=0.0174532925199433;
  BEGIN
    CS:=COS(X*RAD)
  END;
(*-----------------------------------------------------------*)
(* TN: Тангенс-функция (градусы)                             *)
(*-----------------------------------------------------------*)
FUNCTION TN(X:REAL):REAL;
  CONST RAD=0.0174532925199433;
  VAR XX: REAL;
  BEGIN
   XX:=X*RAD; TN:=SIN(XX)/COS(XX);
  END;
(*-----------------------------------------------------------*)
(* ASN: Арксинус-функция (градусы)                           *)
(*-----------------------------------------------------------*)
FUNCTION ASN(X:REAL):REAL;
  CONST RAD=0.0174532925199433, EPS=1E-7;
  BEGIN
    IF  ABS(X)=1.0
        THEN ASN:=90.0*X
        ELSE IF (ABS(X)>EPS)
        THEN  ASN:=ARCTAN(X/SQRT((1 0-X)*(1.0+X)))/RAD
                                      ELSE ASN:=X/RAD;
  END;
(*-----------------------------------------------------------*)
(*  ACS: Арккосинус-функция (градусы)                        *)
(*-----------------------------------------------------------*)
FUNCTION ACS(X:REAL):REAL;
  CONST RAD=0.0174532925199433; EPS=1E-7  C=90.0;
  BEGIN
    IF  ABS(X)=1.0
       THEN ACS:=C-X*C
     ELSE IF (ABS(X)>EPS)
         THEN  ACS:=C-ARCTAN(X/SQRT((1.0-X)*(1.0+X)))/RAD
                                      ELSE ACS:=C-X/RAD;
  END;
(*-----------------------------------------------------------*)
(* ATN: Арктангенс-функция (градусы)                         *)
(*-----------------------------------------------------------*)
FUNCTION ATN(X:REAL):REAL,
```

```
 CONST RAD=0 0174532925199433;
 BEGIN
    ATN =ARCTAN(X)/RAD
 END,
(*-------------------------------------------------------------- *)
```

Функция CUBR позволяет вычислить кубический корень ее аргумента. Она понадобится в следующих главах, но мы решили представить ее здесь вместе с остальными функциями

```
(*-------------------------------------------------------------- *)
(* CUBR Корень кубический                                          *)
(*-------------------------------------------------------------- *)
FUNCTION CUBR(X REAL) REAL,
 BEGIN
   IF (X=0 0)    THEN CUBR =0 0   ELSE CUBR =EXP(LN(X)/3 0)
 END,
(*-------------------------------------------------------------- *)
```

Один из вариантов функции ATN, известный в языке FORTRAN как ATN2, служит для вычисления арктангенса дроби y/x, исключая значение $x = 0$, когда дробь становится неопределенной

```
(*-------------------------------------------------------------- *)
(* ATN2 Арктангенс y/x для двух аргументов                         *)
(*  (с учетом квадранта, -180 град <= ATN2 <= +180 град )          *)
(*-------------------------------------------------------------- *)
FUNCTION ATN2(Y,X REAL) REAL,
 CONST RAD=0 0174532925199433,
 VAR    AX,AY,PHI  REAL,
 BEGIN
   IF (X=0 0) AND (Y=0 0)
        THEN ATN2 =0 0
        ELSE
          BEGIN
               AX =ABS(X),    AY =ABS(Y),
          IF (AX>AY)
               THEN PHI =ARCTAN(AY/AX)/RAD
               ELSE PHI =90 0-ARCTAN(AX/AY)/RAD
          IF (X<0 0) THEN PHI =  180 0-PHI,
          IF (Y<0 0) THEN PHI =  -PHI,
          ATN2.=PHI,
        END,
 END,
(*-------------------------------------------------------------- *)
```

Эта функция используется нами для пересчета прямоугольных координат точки на плоскости (x, y) в полярные координаты r и φ В то время как ATN(y/x) дает в результате угол между -90 и +90° (I и IV квадрант), в результате выполнения ATN2(y/x) угол φ, соответствующий x и y, в случае отрицательного x лежит между +90 и +270° Однако углы от 180 до 360° (как и в FORTRAN'е) заменяются отрицательными, отсчитываемыми от −180 до 0°

Немного сложнее выглядит преобразование пространственных прямоугольных координат (x, y, z) в полярные (r, ϑ, φ) Между ними существует зависимость (рис 2 1)

$$x = r\cos\vartheta\cos\varphi, \qquad r = \sqrt{x^2+y^2+z^2},$$

$$y = r\cos\vartheta\sin\varphi, \qquad \operatorname{tg}\varphi = y/x, \tag{2.1}$$

$$z = r\sin\vartheta, \qquad \operatorname{tg}\vartheta = z/\sqrt{x^2+y^2},$$

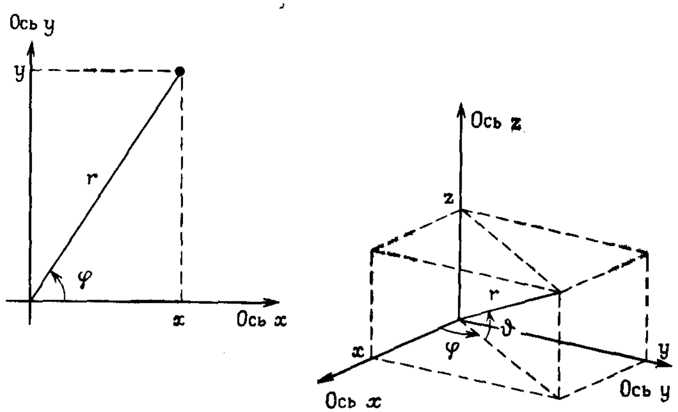


Рис. 2.1. Полярные **координаты** на плоскости и в пространстве

Для выполнения этого очень часто используемого преобразования служат подпрограммы CART и POLAR Процедурой POLAR можно пользоваться и в исключительном случае, когда x и y одновременно равны нулю При этом значение φ (произвольное) предполагается равным нулю В зависимости от знака z значение ϑ равно −90°, 0 или +90°

```
(*---------------------------------------------------------------------*)
(* CART  Преобразование полярных координат (r,theta,phi)               *)
(*           в прямоугольные координаты (x,y,z)                        *)
(*  ( theta = [-90 град,+90 град], phi = [-360 град,+360 град] )       *)
(*---------------------------------------------------------------------*)
PROCEDURE CART(R,THETA,PHI REAL, VAR X,Y,Z. REAL),
 VAR RCST = REAL,
 BEGIN
   RCST    = R*CS(THETA),
```

```
 X  := RCST*CS(PHI);  Y := RCST*SN(PHI);  Z := R*SN(THETA)
 END;
(*-----------------------------------------------------------------------*)
(* POLAR: Преобразование прямоугольных координат (x,y,z)                 *)
(*          в полярные (r,theta,phi)                                     *)
(*   (theta = [-90 град, +90 град]; phi = [-180 град, +180 град])        *)
(*-----------------------------------------------------------------------*)
 PROCEDURE POLAR(X,Y,Z:REAL;VAR R,THETA,PHI:REAL);
  VAR RHO: REAL;
  BEGIN
    RHO: =X*X+Y*Y;        R:=SQRT(RHO+Z*Z);
    PHI: =ATN2(Y,X);      IF PHI<0 THEN PHI:=PHI+360.0;
    RHO: =SQRT(RHO);      THETA:=ATN2(Z,RHO);
 END;
(*-----------------------------------------------------------------------*)
```

Когда величину угла задают в градусах, то доли градуса задаются в минутах и секундах дуги, что соответствует исчислению времени в минутах и секундах. Преобразование одного представления в другое бывает нужно не более одного раза при вводе или выводе определенной величины. Пока все значения величины положительны, такое преобразование не представляет никаких трудностей. Однако при отрицательных значениях углов нужно быть очень внимательным, чтобы правильно учесть знак. Действие процедур GGG и GMS проще всего пояснить примерами:

GG	G	M	S
15.50000	15	30	00.0
-8 15278	-8	09	10.0
0 01667	0	1	0.0
-0.08334	0	-5	0.0

При преобразовании отрицательного числа GG в градусы, минуты и секунды знак минус будет стоять перед первой из величин G, M или S, имеющей ненулевое значение. Нужно еще отметить, что в обеих процедурах величины G и M могут принимать только целые значения и поэтому они определяются как INTEGER, в то время как S описывается как REAL.

```
(*-----------------------------------------------------------------------*)
(* GGG.Преобразование градусов, минут и секунд в градусы с долями*)
(*-----------------------------------------------------------------------*)
 PROCEDURE GGG(G,M:INTEGER;S.REAL;VAR GG:REAL);
  VAR SIGN: REAL;
 BEGIN
   IF ( (G<0) OR (M<0) OR (S<0) ) THEN SIGN:=-1.0 ELSE SIGN:=1.0;
   GG:=SIGN*(ABS(G)+ABS(M)/60.0+ABS(S)/3600 0);
 END;
(*-----------------------------------------------------------------------*)
(* GMS:Преобразование градусов с долями в градусы,минуты,секунды         *)
(*-----------------------------------------------------------------------*)
```

```
PROCEDURE GMS(GG:REAL;VAR G,M:INTEGER;VAR S:REAL);
VAR G1:REAL;
BEGIN
   G1:=ABS(GG);         G: =TRUNC(G1);
   G1:=(G1-G)*60.0,     M: =TRUNC(G1);   S:=(G1-M)*60.0;
   IF (GG<0) THEN
     IF (G<>0) THEN G: =-G ELSE IF (M<>0) THEN M:=-M ELSE S:=-S;
END;
(*-----------------------------------------------------------------*)
```

2.2. Календарь и юлианские даты

При вычислении эфемерид часто бывает нужно вычислить разность двух дат. При этом очень удобным оказывается непрерывный счет дней, который с давних пор практикуется в астрономии. Юлианской датой называется число суток, прошедших с 1 января 4713 г. до нашей эры. Это исчисление названо именем Юлиуса Скалигера, отца Жозефа Юстуса Скалигера, впервые использовавшего такое исчисление. Поскольку счет юлианских дней начинается с библейских времен, современные юлианские даты выражаются большими числами Например, в полдень 23.7.1980 число юлианских дней равнялось 2444444,0. Учитывая, что секунда составляет примерно одну стотысячную суток, для ввода точного момента, выраженного в юлианских днях, требуется 12-разрядное число. Поскольку на протяжении почти трех столетий первые три цифры не меняются, наряду с истинными юлианскими датами используют так называемые *модифицированные юлианские даты*:

$$MJD = JD - 2400000{,}5.$$

Здесь исчисление начинается с 0 часов 17.11.1858 г. При этом момент изменения даты — привычная полночь, а не полдень, как в случае юлианских дат.

```
(*-----------------------------------------------------------------*)
(* MJD: Модифицированная юлианская дата                             *)
(*      для любой даты с 4713 г до нашей эры                         *)
(*      по юлианскому календарю до 4 октября 1582 г.                 *)
(*      по григорианскому календарю с 15 октября 1582 г.             *)
(*-----------------------------------------------------------------*)
FUNCTION MJD(DAY,MONTH,YEAR:INTEGER,HOUR:REAL):REAL,
  VAR A: REAL; B: INTEGER;
  BEGIN
     A:=10000.0*YEAR+100 0*MONTH+DAY;
    IF (MONTH<=2) THEN BEGIN MONTH:=MONTH+12; YEAR:=YEAR-1 END;
    IF    (A<=15821004.1)
       THEN B:=-2+TRUNC((YEAR+4716)/4)-1179
       ELSE B:=TRUNC(YEAR/400)-TRUNC(YEAR/100)+TRUNC(YEAR/4);
```

```
    A  :  =365.0*YEAR-679004 0,
    MJD: =A+B+TRUNC(30.6001*(MONTH+1))+DAY+HOUR/24.0;
 END,
(*-----------------------------------------------------------------      *)
```

Например:

```
 VAR HOUR, MODJD,JD· REAL;
 BEGIN
 GGG(3,30,10.0,HOUR), (* Дата 14 янв  1961 г., 3 ч 30 мин 10 сек   *)
 MODJD:= MJD(14,1,1961,HOUR),  (* Модиф  юлианская дата            *)
    JD· = MODJD + 2400000.5,   (* юлианская дата                   *)
    WRITELN('   14 Jan.1961,    3h30m10s '),
    WRITELN(' MJD=  ',  MODJD 20 5),
    WRITELN(' JD  = ',        JD 20·5),
 END.
```

В MJD учтено, что, согласно григорианской реформе, после 4 октября 1582 г (JD 2299159,5) следует 15 октября (JD 2299160,5) До этого времени действовал юлианский календарь, по которому в каждом четвертом году добавлялось 29 февраля Со времени реформы этот добавочный день не содержится в «круглых» годах столетия в том случае, если число лет не делится на 400 Средняя продолжительность григорианского года составляет

$$365 + 1/4 - 1/100 + 1/400 = 365,2425 \text{ сут},$$

что немного короче юлианского года, равного 365,25 сут

Программа CALDAT позволяет по модифицированным юлианским датам вычислить привычные календарные даты Она представлена здесь, поскольку имеет прямое отношение к обсуждаемым вопросам.

```
(*-----------------------------------------------------------------      *)
(* CALDAT·     Определение календарных дат                                *)
(*             по модифицированным юлианским датам (MJD)                  *)
(*-----------------------------------------------------------------      *)
 PROCEDURE CALDAT(MJD·REAL,
   VAR DAY,MONTH,YEAR·INTEGER,VAR HOUR REAL);
   VAR B,D,F         INTEGER,
        JD,JD0,C,E.  REAL,
   BEGIN
     JD    := MJD + 2400000 5;
     JD0   := TRUNC(JD+0 5),              (* Standard Pascal        *)
     (* JD0:= INT(JD+0.5),          *)    (* TURBO Pascal           *)
     (* JD0:= LONG_TRUNC(JD+0.5),*)       (* ST Pascal plus         *)
     IF  (JD0<2299161 0)                  (* календарь·             *)
        THEN BEGIN B =0, C:=JD0+1524 0 END    (* -> юлианский       *)
        ELSE  BEGIN                  (* -> григорианский  *)
              B:=TRUNC((JD0-1867216 25)/36524.25);
              C:=JD0+(B-TRUNC(B/4))+1525.0
```

```
          END;
    D    : = TRUNC((C-122 1)/365 25);
    E    : = 365 0*D+TRUNC(D/4);
    F    : =  TRUNC((C-E)/30 6001);
    DAY  : = TRUNC(C-E+0 5)-TRUNC(30 6001*F);
    MONTH:=  F-1-12*TRUNC(F/14);
    YEAR : = D-4715-TRUNC((7+MONTH)/10);
    HOUR: =    24.0*(JD+0 5-JD0);
  END;
(*---------------------------------------------------------------------*)
```

2.3. Эклиптика и экватор

Эклиптические и экваториальные координаты отличаются друг от друга, так как измеряются относительно разных плоскостей. В одном случае плоскость $x - y$ — это плоскость орбиты Земли (эклиптика), в другом — плоскость, перпендикулярная земной оси и параллельная плоскости земного экватора (рис 2 2) Линия пересечения этих плоскостей определяет направление на точку весеннего равноденствия (♈). Оно перпендикулярно направлениям на Северный полюс эклиптики (ось z) и Северный полюс мира (ось z') Угол между эклиптикой и экватором составляет около $23{,}5°$

Как выглядит зависимость между эклиптическими координатами (x, y, z) и экваториальными координатами (x', y', z') точки? Поскольку как $x = x'$, достаточно рассмотреть преобразование $(y, z) \Leftrightarrow (y', z')$. Прежде всего заметим, что точка на оси y ($z = 0$) имеет экваториальные координаты $y' = +y \cos \varepsilon$ и $z' = +y \sin \varepsilon$ Если точка лежит на оси z, ее экваториальные координаты будут $y' = -z \sin \varepsilon$ и $z' = +z \cos \varepsilon$ Обобщая, получаем, что любая точка в экваториальной системе имеет координаты

$$\begin{aligned} x' &= +x, \\ y' &= +y \cos \varepsilon - z \sin \varepsilon, \\ z' &= +y \sin \varepsilon + z \cos \varepsilon. \end{aligned} \tag{2.2}$$

Точно так же можно получить и обратное преобразование:

$$\begin{aligned} x &= +x', \\ y &= +y' \cos \varepsilon + z' \sin \varepsilon, \\ z &= -y' \sin \varepsilon + z' \cos \varepsilon \end{aligned} \tag{2.3}$$

Полярные координаты, соответствующие (x, y, z), называются эклиптической долготой l, эклиптической широтой b и расстоянием r (рис 2 3) В экваториальной системе этим координатам соответствуют прямое восхождение α, склонение δ и расстояние r, которое в обеих

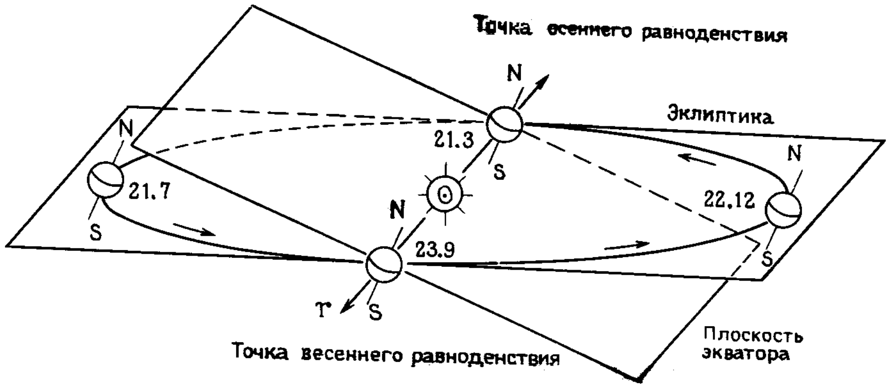


Рис. 2.2. Эклиптика и экватор.

системах имеет одинаковые значения:

$$
\begin{aligned}
x &= r\cos b\cos l, & x' &= r\cos\delta\cos\alpha,\\
y &= r\cos b\sin l, & y' &= r\cos\delta\sin\alpha, \qquad (2.4)\\
z &= r\sin b, & z' &= r\sin\delta.
\end{aligned}
$$

Из этих формул видно, что с помощью подпрограммы POLAR прямоугольные координаты можно преобразовать в долготы, широты и расстояния, а также в прямое восхождение и склонение. Рассмотрим небольшой пример программы:

```
   VAR X,Y,Z,R,B,L,EPS,CEPS,SEPS,
   XSTRICH,YSTRICH,ZSTRICH,ALPHA,DELTA: REAL;
 BEGIN
   READ(X,Y,Z);
   POLAR(X,Y,Z,R,B,L);
   WRITELN(' Долгота: ',L,' Широта: ',B,' Расстояние: ',R);
   EPS     :=23.5; CEPS:=CS(EPS); SEPS:=SN(EPS);
   XSTRICH := X;
   YSTRICH := CEPS*Y-SEPS*Z;
   ZSTRICH :=SEPS*Y +CEPS*Z;
   POLAR(XSTRICH,YSTRICH,ZSTRICH,R,DELTA,ALPHA);
   WRITELN(' Прямое восхождение: ',ALPHA/15.0,' Склонение:',DELTA,
           'Расстояние: ',R);
 END.
```

Значения прямого восхождения при этом будут выражены, как обычно принято, в часах ($1^h \equiv 15°$), а не в градусах.

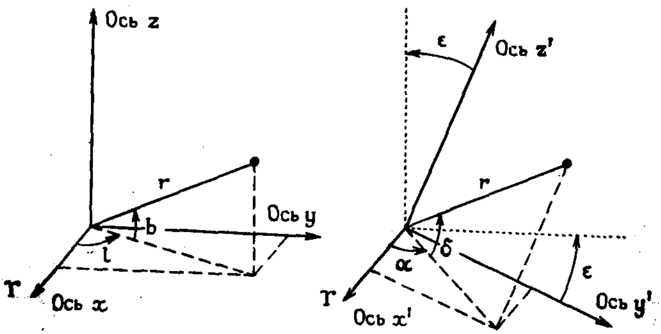


Рис.2.3. Эклиптические и экваториальные координаты.

До сих пор мы не говорили о точном значении угла наклона эклиптики ε. Это значение не постоянно, и уменьшается примерно на 47″ за 100 лет. Это происходит из-за медленного движения орбиты Земли, обусловленного силой тяготения других планет. При правильном преобразовании координат важно принять во внимание, к какой эклиптике относятся координаты—1950 или 2000 гг. Эти годы называют равноденствием координат. Точное значение угла наклона эклиптики как функции времени таково:

$$\varepsilon = 23{,}43929111^\circ - 46{,}8150'' \, T - 0{,}00059'' \, T^2 + 0{,}001813'' \, T^3, \qquad (2.5)$$

где T—число юлианских столетий между принятым равноденствием и полуднем 1 января 2000 г. В отличие от григорианского века, который в среднем равен 36524,25 сут, юлианский век всегда точно равен 36525 сут. Наиболее важными считаются эпохи J1900 (0,5 января 1900, JD 2415020,0) и J2000 (1,5 января 2000, JD 2451545,0). Между ними как раз 100 юлианских лет Для равноденствия с юлианской датой JD значение T вычисляют по формуле

$$T = \frac{\mathrm{JD} - 2451545}{36525}.$$

Приближенно вычисляют T для года y так. $T \simeq (y - 2000)/100$.

```
(*------------------------------------------------------------------*)
(* ECLEQU: Преобразование эклиптических коорд. в экваториальные     *)
(*         (T: равноденствие в юлианских столетиях с J2000)         *)
(*------------------------------------------------------------------*)
PROCEDURE ECLEQU(T:REAL;VAR X,Y,Z:REAL);
  VAR EPS,C,S,V: REAL;
  BEGIN
    EPS: =23.43929111-(46.8150+(0.00059-0.001813*T)*T)*T/3600.0;
    C  : =CS(EPS);      S:=SN(EPS);
    V  : =+C*Y-S*Z;     Z:=+S*Y+C*Z,      Y:=V;
  END;
(*------------------------------------------------------------------*)
(* EQUECL: Преобразование экваториальных коорд. в эклиптические     *)
(*         (T: равноденствие в юлианских столетиях с J2000)         *)
(*------------------------------------------------------------------*)
PROCEDURE EQUECL(T:REAL;VAR X,Y,Z:REAL);
  VAR EPS,C,S,V: REAL;
  BEGIN
    EPS: =23.43929111-(46.8150+(0.00059-0.001813*T)*T)*T/3600.0;
    C  : =CS(EPS);      S:=SN(EPS);
    V  : =+C*Y+S*Z;     Z:=-S*Y+C*Z;      Y:=V;
  END;
(*------------------------------------------------------------------*)
```

Подпрограммы ECLEQU и EQUECL построены таким образом, что обеспечивают взаимное преобразование координат Вызывая одну из них вслед за другой (с одним значением *T*), получаем исходные данные.

Приведенный выше пример при помощи ECLEQU может быть переписан следующим образом (равноденствие J1950 = JD 2433282.50):

```
CONST JD2000 = 2451545.0;
      JD1950 = 2433283 5;
VAR X,Y,Z,R,B,L,
    T,DELTA,ALPHA: REAL;
BEGIN
  READ(X,Y,Z);
  POLAR(X,Y,Z,R,B,L);
  WRITELN(' Долгота: ',L,' Широта: ',B,' Расстояние: ' ,R);
  T:= (JD1950-JD2000) / 36525;
  ECLEQU(T,X,Y,Z);
  POLAR(X,Y,Z,R,DELTA,ALPHA);
  WRITELN(' Прямое восхождение: ',  ALPHA/15.0,
          ' Склонение: ',DELTA,' Расстояние: ', R);
END.
```

2.4. Прецессия

Перемещение земной оси и эклиптики под влиянием силы тяготения Солнца, Луны и планет ведет не только к изменению угла между экватором и эклиптикой, но и к смещению точки равноденствия примерно на 1,5° за 100 лет (1′ в год). Поэтому для точных вычислений обязательно нужно указывать равноденствие используемой системы координат. Обычно используют

равноденствие даты,
равноденствие J2000 и
равноденствие B1950

Равноденствие даты означает, что координаты вычисляются относительно положения эклиптики и точки весеннего равноденствия на заданную дату. Подобное ежедневное изменение системы координат имеет смысл, если, например, координаты планеты нужны для работы на разделенном круге телескопа с паралактической установкой или на меридианном круге. Из-за перемещения земной оси меняется и ориентация часовой оси телескопа. При исследовании истинного пространственного движения планет лучше работать с фиксированным равноденствием, например, с юлианской эпохой J2000 (1,5 января 2000 = JD 2451545,0), которая введена повсеместно с 1984 г. Прежде использовалась старая эпоха равноденствия B1950, применявшаяся во многих звездных каталогах и атласах (например SAO Star Catalogue и Atlas Coeli). Буква B означает, что эта эпоха относится не к середине эпох J1900 и J2000 (которая пришлась бы на J1950 = JD 2433282,5), а к началу бесселева 1950 г. (0,923 января 1950 = JD 2433282,423).

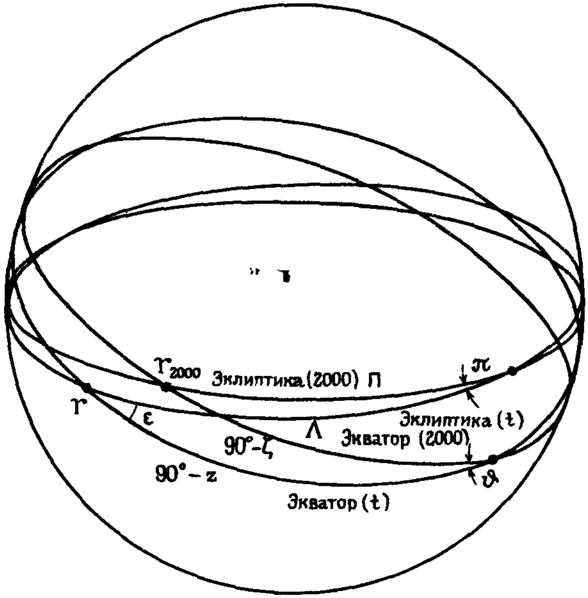


Рис.2.4. Влияние прецессии на эклиптику, экватор и точку весеннего равноденствия.

Преобразование координат от одного равноденствия к другому выполняется так же, как и преобразование эклиптических координат в экваториальные. Но если системы координат преобразуются одна в другую простым поворотом вокруг оси x, то при преобразовании координат от одного равноденствия к другому требуется вращение по всем трем осям x, y и z

Рассмотрим положение эклиптики в момент T_0 и в момент $T_0 + T$. Положения эклиптики в эти моменты составляют между собой угол

$$\pi = (47{,}0029'' - 0{,}06603''\,T_0 + 0{,}000598''\,T_0^2)\,T + (-0{,}03302'' + 0{,}000598''\,T_0)T^2 + 0{,}000060''\,T^3. \tag{2.6}$$

Предположим, что мы имеем две системы координат (x', y', z') и (x'', y'', z''). Плоскость $x' - y'$ одной из них совпадает с эклиптикой в момент T_0, а плоскость $x'' - y''$ другой — в момент $T_0 + T$ Тогда получим:

$$x'' = x',$$

$$y'' = +\cos \pi \cdot y' + \sin \pi \cdot z',$$

$$z'' = -\sin \pi \cdot y' + \cos \pi \cdot z'.$$

При этом предполагается, что оси x' и x'' ориентированы в направлении линии пересечения обеих плоскостей. Теперь предположим, что (x_0, y_0, z_0) — координаты равноденствия T_0, а (x, y, z) — координаты, соответствующие равноденствию $T_0 + T$ Имеем

$$x' = +\cos \Pi \cdot x_0 + \sin \Pi \cdot y_0,$$

$$y' = -\sin \Pi \cdot x_0 + \cos \Pi \cdot y_0,$$

$$z' = z_0$$

и

$$x = +\cos \Lambda \cdot x'' - \sin \Lambda \cdot y'',$$

$$y = +\sin \Lambda \cdot x'' + \cos \Lambda \cdot y'',$$

$$z = z'',$$

где

$$\Pi = (174{,}876383889^\circ + 3289{,}4789'' T_0 + 0{,}60622'' T_0^2) + (-869{,}8089'' - 0{,}50491'' T_0)\, T + 0{,}03536'' T^2,$$

$$p = (5029{,}0966'' + 2{,}22226'' T_0 - 0{,}000042'' T_0^2)\, T + (1{,}11113'' - 0{,}000042'' T_0) T^2 - 0{,}000006'' T^3, \quad (2.7)$$

$$\Lambda = \Pi + p.$$

Здесь Π и Λ — углы между осями x'/x'' и точкой равноденствия Υ_0 для эпохи T_0 (ось x_0) или точкой Υ в момент $T_0 + T$ (ось x), а p называется прецессией по долготе, так как в основном представляет перемещение точки весеннего равноденствия по эклиптической долготе.

Подставляя эти выражения одно в другое, окончательно получаем

$$x = a_{11}\,x_0 + a_{12}\,y_0 + a_{13}\,z_0,$$
$$y = a_{21}\,x_0 + a_{22}\,y_0 + a_{23}\,z_0, \qquad (2.8)$$
$$z = a_{31}\,x_0 + a_{32}\,y_0 + a_{33}\,z_0,$$

где

$$a_{11} = +\cos\Lambda\cos\Pi + \sin\Lambda\cos\pi\sin\Pi,$$
$$a_{21} = +\sin\Lambda\cos\Pi - \cos\Lambda\cos\pi\sin\Pi,$$
$$a_{31} = +\sin\pi\sin\Pi,$$

$$a_{12} = +\cos\Lambda\sin\Pi - \sin\Lambda\cos\pi\cos\Pi,$$
$$a_{22} = +\sin\Lambda\sin\Pi + \cos\Lambda\cos\pi\cos\Pi, \qquad (2.9)$$
$$a_{32} = -\sin\pi\cos\Pi,$$

$$a_{13} = -\sin\Lambda\sin\pi,$$
$$a_{23} = +\cos\Lambda\sin\pi,$$
$$a_{33} = +\cos\pi.$$

В экваториальной системе координат трем углам π, Π и Λ соответствуют углы $90^\circ - \zeta$, ϑ и $90^\circ + z$:

$$\zeta = (2306{,}2181'' + 1{,}39656''\,T_0 - 0{,}000139''\,T_0^2)\,T + (0{,}30188'' - 0{,}000345''\,T_0)\,T^2 + 0{,}017998''\,T^3,$$
$$\vartheta = (2004{,}3109'' - 0{,}85330''\,T_0 - 0{,}000217''\,T_0^2)\,T + (-0{,}42665'' - 0{,}000217''\,T_0)\,T^2 - 0{,}041833''\,T^3, \qquad (2.10)$$
$$z = \zeta + (0{,}79280'' + 0{,}000411''\,T_0)T^2 + 0{,}000205''\,T^3.$$

Поэтому в соответствующих формулах

$$a_{11} = -\sin z\sin\zeta + \cos z\cos\vartheta\cos\zeta,$$
$$a_{21} = +\cos z\sin\zeta + \sin z\cos\vartheta\cos\zeta,$$
$$a_{31} = +\sin\vartheta\cos\zeta,$$

$$a_{12} = -\sin z\cos\zeta - \cos z\cos\vartheta\sin\zeta,$$
$$a_{22} = +\cos z\cos\zeta - \sin z\cos\vartheta\sin\zeta, \qquad (2.11)$$
$$a_{32} = -\sin\vartheta\sin\zeta,$$

$$a_{13} = -\cos z\sin\vartheta,$$
$$a_{23} = -\sin z\sin\vartheta,$$
$$a_{33} = +\cos\vartheta.$$

При вычислении прецессии наиболее трудоемким является определение углов и значений a_{ij}. Однако эти величины зависят только о двух эпох T_0 и $T_0 + T$ и могут быть повторно использованы, если целый ряд положений нужно перевести из одной эпохи в другую. Соответствующие стадии вычисления оформлены как отдельные подпрограммы.

```
(*-------------------------------------------------------------------*)
(* PMATECL: Вычисление прецессионной  матрицы A[i,j] для           *)
(*           преобразования                                          *)
(*           эклиптических координат от равноденствия T1 к T2        *)
(*           (T=(JD-2451545.0)/36525)                                *)
(*-------------------------------------------------------------------*)
PROCEDURE PMATECL(T1,T2:REAL;VAR A: REAL33);
  CONST SEC=3600.0,
  VAR DT,PPI,PI,PA: REAL;
      C1,S1,C2,S2,C3,S3:   REAL;
  BEGIN
    DT : =T2-T1;
    PPI: =  174 876383889  +(  ((3289.4789+0.60622*T1)*T1)  +
              ((-869 8089-0.50491*T1)  +  0.03536*DT)*DT   )/SEC;
     PI: =  (   (47.0029-(0.06603-0.000598*T1)*T1)+
              ((-0.03302+0.000598*T1)+0 000060*DT)*DT      )*DT/SEC;
     PA: =      ((5029.0966+(2.22226-0.000042*T1)*T1)+
              ((1.11113-0.000042*T1)-0.000006*DT)*DT       )*DT/SEC;
     C1: =CS ( PPI+PA);   C2.=CS(PI);     C3:=CS(PPI);
     S1: = SN(PPI+PA);    S2:=SN(PI);     S3:=SN(PPI);
  A[1,1]: =+C1*C3+S1*C2*S3;  A[1,2].=+C1*S3-S1*C2*C3;    A[1,3]:=-S1*S2;
  A[2,1]: =+S1*C3-C1*C2*S3;  A[2,2]:=+S1*S3+C1*C2*C3;    A[2,3]:=+C1*S2;
  A[3,1]: =+S2*S3;           A[3,2]:=-S2*C3;             A[3,3]:=+C2;
END;
(*-------------------------------------------------------------------*)
(* PMATEQU: Вычисление прецессионной матрицы A[i,j] для             *)
(*          преобразования                                           *)
(*          экваториальных координат от равноденствия T1 к T2        *)
(*          (T=(JD-2451545.0)/36525)                                 *)
(*-------------------------------------------------------------------*)
PROCEDURE PMATEQU(T1,T2:REAL;VAR A:REAL33);
  CONST SEC=3600.0;
  VAR DT,ZETA,Z,THETA: REAL;
      C1,S1,C2,S2,C3,S3:   REAL;
   BEGIN
  DT    : =T2-T1;
  ZETA : =       ((2306.2181+(1.39656-0.000139*T1)*T1)+
             ((0.30188-0.000345*T1)+0.017998*DT)*DT       )*DT/SEC;
  Z     : =  ZETA  +  (  (0.79280+0.000411*T1)+0.000205*DT)*DT*DT/SEC;
  THETA: =       ((2004.3109-(0.85330+0.000217*T1)*T1)-
             ((0.42665+0.000217*T1)+0.041833*DT)*DT       )*DT/SEC;
  C1    : =CS(Z);    C2: =CS(THETA);   C3: =CS(ZETA);
  S1    : =SN(Z);    S2: =SN(THETA);   S3: =SN(ZETA);
 A[1,1]: =-S1*S3+C1*C2*C3;  A[1,2]:=-S1*C3-C1*C2*S3;     A[1,3]:=-C1*S2;
```

```
A[2,1]:  =+C1*S3+S1*C2*C3;  A[2,2]:=+C1*C3-S1*C2*S3;  A[2,3]:=-S1*S2;
A[3,1]:  =+S2*C3;           A[3,2]:=-S2*S3;           A[3,3]:=+C2;
END;
(*---------------------------------------------------------------*)
(* PRECART: Преобразование прецессии с известной матрицей A[i,j] *)
(*          для эклиптических и экваториальных координат         *)
(*          (применяется вместе с PMATECL и PMATEQU)              *)
(*---------------------------------------------------------------*)
PROCEDURE PRECART(A:REAL33; VAR X,Y,Z:REAL);
  VAR U,V,W: REAL;
  BEGIN
    U: =    A[1,1]*X+A[1,2]*Y+A[1,3]*Z;
    V: =    A[2,1]*X+A[2,2]*Y+A[2,3]*Z;
    W: =    A[3,1]*X+A[3,2]*Y+A[3,3]*Z;
    X: = U;  Y:=V;  Z:=W;
  END;
(*---------------------------------------------------------------*)
```

Матрица A принадлежит к типу данных REAL33. Соответствующее общее определение типа

```
TYPE REAL33 = ARRAY[1..3,1..3] OF REAL;
```

вставляется в описательную часть программы.

Чтобы наглядно представить влияние прецессии и потренироваться в использовании различных программ, приведем довольно длинный пример, в котором эклиптические координаты равноденствия B1950 преобразуются в соответствующие экваториальные координаты J2000. Это можно сделать двумя путями: либо вычислить сначала прецессию, либо вначале выполнить преобразование эклиптика — экватор.

```
CONST B1950 = -0.500002108; (* B1950=(2433282.423-2451545)/36525   *)
      J2000 = 0.0;
VAR   A                      : REAL33;
      R,B,L,X0,Y0,Z0,
      X,Y,Z,DEC,RA           : REAL;
      I                      : INTEGER;
BEGIN

  (* эклиптические координаты B1950 (градусы):    *)
  L :=200.0;  B:=10.0;  R:=1.0;
  CART (R,B,L,X0,Y0,Z0);

  (* способ 1:        *)
  PMATECL(B1950,J2000,A); (* матрица B1950->J2000 эклиптич.;        *)
  FOR I:=1 TO 3 DO WRITELN(A[I,1]:15:10,A[I,2]:15:10,A[I,3]:15:10);
  X:=X0;  Y:=Y0;  Z:=Z0;            (* экл.    B1950     *)
  PRECART(A,X,Y,Z);                 (* экл.    J2000     *)
```

```
  POLAR(X,Y,Z,R,B,L);
  WRITELN(L:15:10,B:15:10,R:15:10);

  (* способ 2: *)
  PMATEQU(B1950,J2000,A); (* матрица B1950->J2000 экваториал.:     *)
  FOR I:=1 TO 3 DO WRITELN(A[I,1]:15:10,A[I,2]:15:10,A[I,3]:15:10);
  X:=X0; Y:=Y0; Z:=Z0;       (* экл. B1950   *)
  ECLEQU(B1950,X,Y,Z);       (* экв. B1950   *)
  POLAR(X,Y,Z,R,DEC,RA);
  WRITELN(RA/15:15:10,DEC:15:10,R:15:10);
  PRECART(A,X,Y,Z);          (* экв. J2000   *)
  POLAR(X,Y,Z,R,DEC,RA);
  WRITELN(RA/15:15:10,DEC:15:10,R:15:10);
END.
```

2.5. Геоцентрические координаты и путь Солнца

Для полноты программы здесь не хватает возможности взаимного преобразования гелиоцентрических (относительно центра Солнца) и геоцентрических (относительно центра Земли) координат. Это преобразование начала координат описывается равенствами

$$\mathbf{r}_p = \mathbf{r}_{\odot p} + \mathbf{r}_{\odot} \text{ и } \mathbf{r}_{\odot p} = \mathbf{r}_p - \mathbf{r}_{\odot}, \tag{2.12}$$

которые получаются из треугольника, представленного на рис. 2.5.

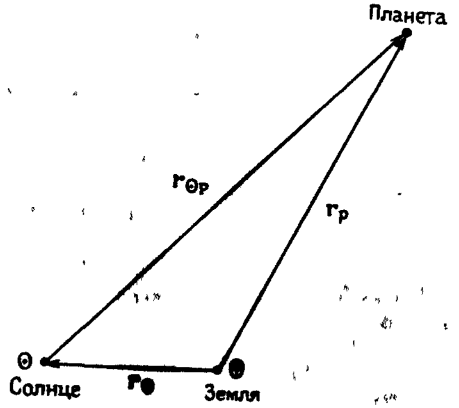


Рис. 2.5. Треугольник Земля—Солнце—планета.

Здесь $\mathbf{r}_{\odot p}$ и $\mathbf{r}_p$ — соответственно векторы гелиоцентрического и геоцентрического положений точки P, а $\mathbf{r}_\odot$ — геоцентрическое положение Солнца. Компоненты формул преобразования выглядят так:

$$\begin{aligned} x_p &= x_{\odot p} + x_\odot, \\ y_p &= y_{\odot p} + y_\odot \\ z_p &= z_{\odot p} + z_\odot, \end{aligned} \quad \text{и} \quad \begin{aligned} x_{\odot p} &= x_p - x_\odot, \\ y_{\odot p} &= y_p - y_\odot, \\ z_{\odot p} &= z_p - z_\odot, \end{aligned}$$

где неважно, какие именно координаты, эклиптические или экваториальные, использованы для представления различных векторов.

Поскольку любая ошибка в положении Солнца неизбежно сказывается в этом случае на точности вычислений, стоит приложить некоторые усилия для вычисления координат Солнца. Приведенная здесь процедура SUN200 дает точность около 1″, что достаточно для решения многих задач. С помощью этой процедуры для каждого момента

$$T = (\mathrm{JD} - 2451545)/36525$$

вычисляется эклиптическая долгота (L) и широта (B) Солнца, а также его расстояние от Земли (R). По этим данным определяются прямоугольные эклиптические координаты:

$$\begin{aligned} x_\odot &= R \cos B \cos L, \\ y_\odot &= R \cos B \sin L, \\ z_\odot &= R \sin B. \end{aligned}$$

Здесь подразумевается, что L и B имеют равноденствие даты, но, используя приведенные выше формулы, можно привести L и B к любому другому равноденствию.

```
(*-----------------------------------------------------------------*)
(* SUN200: Солнце; эклиптические координаты L,B,R (в градусах     *)
(*         и а.е.) на равноденствие даты                           *)
(*         (T: время в юлианских столетиях с J2000)                *)
(*         (= (JED-2451545.0)/36525)                               *)
(*-----------------------------------------------------------------*)
PROCEDURE SUN200(T:REAL;VAR L,B,R:REAL);
  CONST P2=6.283185307;
  VAR C3,S3            : ARRAY [-1..7] OF REAL;
      C,S              : ARRAY [-8..0] OF REAL;
      M2,M3,M4,M5,M6: REAL;
      D,A,UU           : REAL;
      U,V,DL,DR,DB     : REAL;
      I                : INTEGER;

  FUNCTION FRAC(X:REAL):REAL;
```

```
  (* TRUNC заменить для T<-24 на LONG_TRUNC или INT !          *)
 BEGIN  X:=X-TRUNC(X); IF (X<0) THEN X:=X+1.0; FRAC:=X   END;
PROCEDURE ADDTHE(C1,S1,C2,S2:REAL; VAR C,S:REAL);
  BEGIN C:=C1*C2-S1*S2;  S:=S1*C2+C1*S2;  END;

PROCEDURE TERM(I1,I,IT:INTEGER;DLC,DLS,DRC,DRS,DBC,DBS:REAL);
  BEGIN
    IF  IT=0  THEN  ADDTHE(C3[I1],S3[I1],C[I],S[I],U,V)
             ELSE BEGIN U:=U*T; V:=V*T END;
    DL:=DL+DLC*U+DLS*V; DR:=DR+DRC*U+DRS*V; DB:=DB+DBC*U+DBS*V;
  END;

PROCEDURE PERTVEN;(* кеплеровские члены и возмущения Венерой    *)
  VAR I: INTEGER;
  BEGIN
    C[0]:=1.0;    S[0]:=0.0;    C[-1]:=COS(M2);     S[-1]:=-SIN(M2);
    FOR I:=-1 DOWNTO -5
     DO          ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
   TERM(1,  0, 0, -0.22, 6892.76, -16707.37,  -0.54, 0.00,  0.00);
   TERM(1,  0, 1, -0.06,  -17.35,     42.04,  -0.15, 0.00,  0.00);
   TERM(1,  0, 2, -0.01,   -0.05,      0.13,  -0.02, 0.00,  0.00);
   TERM(2,  0, 0,  0.00,   71.98,   -139.57,   0.00, 0.00,  0.00);
   TERM(2,  0, 1,  0.00,   -0.36,      0.70,   0.00, 0.00,  0.00);
   TERM(3,  0, 0,  0.00,    1.04,     -1.75,   0.00, 0.00,  0.00);
   TERM(0, -1, 0,  0.03,   -0.07,     -0.16,  -0.07, 0.02, -0.02);
   TERM(1, -1, 0,  2.35,   -4.23,     -4.75,  -2.64, 0.00,  0.00);
   TERM(1, -2, 0, -0.10,    0.06,      0.12,   0.20, 0.02,  0.00);
   TERM(2, -1, 0, -0.06,   -0.03,      0.20,  -0.01, 0.01, -0.09);
   TERM(2, -2, 0, -4.70,    2.90,      8.28,  13.42, 0.01, -0.01);
   TERM(3, -2, 0,  1.80,   -1.74,     -1.44,  -1.57, 0.04, -0.06);
   TERM(3, -3, 0, -0.67,    0.03,      0.11,   2.43, 0.01,  0.00);
   TERM(4, -2, 0,  0.03,   -0.03,      0.10,   0.09, 0.01, -0.01);
   TERM(4, -3, 0,  1.51,   -0.40,     -0.88,  -3.36, 0.18, -0.10);
   TERM(4, -4, 0, -0.19,   -0.09,     -0.38,   0.77, 0.00,  0.00);
   TERM(5, -3, 0,  0.76,   -0.68,      0.30,   0.37, 0.01,  0.00);
   TERM(5, -4, 0, -0.14,   -0.04,     -0.11,   0.43,-0.03,  0.00);
   TERM(5, -5, 0, -0.05,   -0.07,     -0.31,   0.21, 0.00,  0.00);
   TERM(6, -4, 0,  0.15,   -0.04,     -0.06,  -0.21, 0.01,  0.00);
   TERM(6, -5, 0, -0.03,   -0.03,     -0.09,   0.09,-0.01,  0.00);
   TERM(6, -6, 0,  0.00,   -0.04,     -0.18,   0.02, 0.00,  0.00);
   TERM(7, -5, 0, -0.12,   -0.03,     -0.08,   0.31,-0.02, -0.01);
 END;

PROCEDURE PERTMAR;(* Возмущения Марсом          *)
  VAR I: INTEGER;
  BEGIN
    C[-1]:=COS(M4);      S[-1]:=-SIN(M4);
```

```
FOR I:=-1 DOWNTO -7
 DO          ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
TERM(1, -1, 0,  -0.22,   0.17,  -0.21,  -0.27,  0.00,  0.00);
TERM(1, -2, 0,  -1.66,   0.62,   0.16,   0.28,  0.00,  0.00);
TERM(2, -2, 0,   1.96,   0.57,  -1.32,   4.55,  0.00,  0.01);
TERM(2, -3, 0,   0.40,   0.15,  -0.17,   0.46,  0.00,  0.00);
TERM(2, -4, 0,   0.53,   0.26,   0.09,  -0.22,  0.00,  0.00);
TERM(3, -3, 0,   0.05,   0.12,  -0.35,   0.15,  0.00,  0.00);
TERM(3, -4, 0,  -0.13,  -0.48,   1.06,  -0.29,  0.01,  0.00);
TERM(3, -5, 0,  -0.04,  -0.20,   0.20,  -0.04,  0.00,  0.00);
TERM(4, -4, 0,   0.00,  -0.03,   0.10,   0.04,  0.00,  0.00);
TERM(4, -5, 0,   0.05,  -0.07,   0.20,   0.14,  0.00,  0.00);
TERM(4, -6, 0,  -0.10,   0.11,  -0.23,  -0.22,  0.00,  0.00);
TERM(5, -7, 0,  -0.05,   0.00,   0.01,  -0.14,  0.00,  0.00);
TERM(5, -8, 0,   0.05,   0.01,  -0.02,   0.10,  0.00,  0.00);
END;

PROCEDURE PERTJUP;(* Возмущения Юпитером     *)
  VAR I: INTEGER;
  BEGIN
    C[-1]:=COS(M5);      S[-1]:=-SIN(M5);
    FOR I:=-1 DOWNTO -3
     DO         ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
    TERM(1, -1, 0,   0.01,   0.07,   0.18,  -0.02,  0.00, -0.02);
    TERM(0, -1, 0,  -0.31,   2.58,   0.52,   0.34,  0.02,  0.00);
    TERM(1, -1, 0,  -7.21,  -0.06,   0.13, -16.27,  0.00, -0.02);
    TERM(1, -2, 0,  -0.54,  -1.52,   3.09,  -1.12,  0.01, -0.17);
    TERM(1, -3, 0,  -0.03,  -0.21,   0.38,  -0.06,  0.00, -0.02);
    TERM(2, -1, 0,  -0.16,   0.05,  -0.18,  -0.31,  0.01,  0.00);
    TERM(2, -2, 0,   0.14,  -2.73,   9.23,   0.48,  0.00,  0.00);
    TERM(2, -3, 0,   0.07,  -0.55,   1.83,   0.25,  0 01,  0.00);
    TERM(2, -4, 0,   0.02,  -0.08,   0.25,   0.06,  0.00,  0.00);
    TERM(3, -2, 0,   0.01,  -0.07,   0.16,   0.04,  0.00,  0.00);
    TERM(3, -3, 0,  -0.16,  -0.03,   0.08,  -0.64,  0.00,  0.00);
    TERM(3, -4, 0,  -0.04,  -0.01,   0.03,  -0.17,  0.00,  0.00);
  END;

PROCEDURE PERTSAT;(* Возмущения Сатурном     *)
  BEGIN
    C[-1]:=COS(M6);      S[-1]:=-SIN(M6);
    ADDTHE(C[-1],S[-1],C[-1],S[-1],C[-2],S[-2]);
    TERM(0, -1, 0,   0.00,   0.32,   0.01,   0.00,  0.00,  0.00);
    TERM(1, -1, 0,  -0.08,  -0.41,   0.97,  -0.18,  0.00, -0.01);
    TERM(1, -2, 0,   0.04,   0.10,  -0.23,   0.10,  0.00,  0.00);
    TERM(2, -2, 0,   0.04,   0.10,  -0.35,   0.13,  0.00,  0.00);
  END;
```

```
PROCEDURE PERTMOO;(* Различие между барицентром системы      *)
                  (* Земля—Луна и центром Земли              *)
  BEGIN
    DL := DL +   6.45*SIN(D)  -  0.42*SIN(D-A)  +  0.18*SIN(D+A)
                             +  0.17*SIN(D-M3)  -  0.06*SIN(D+M3);
    DR := DR + 30.76*COS(D)  -  3.06*COS(D-A)+  0.85*COS(D+A)
                             -0.58*COS(D+M3)  +  0.57*COS(D-M3);
    DB := DB +  0.576*SIN(UU),
  END;

BEGIN (* SUN200  *)

  DL   :=0.0;   DR:=0.0;   DB:=0.0;
  M2   :=P2*FRAC(0.1387306+162.5485917*T);
  M3   :=P2*FRAC(0.9931266+  99.9973604*T);
  M4   :=P2*FRAC(0.0543250+  53.1666028*T);
  M5   :=P2*FRAC(0 0551750+   8.4293972*T);
  M6   :=P2*FRAC(0.8816500+   3.3938722*T);
  D    :=P2*FRAC(0.8274+1236.8531*T);
  A    :=P2*FRAC(0.3749+1325.5524*T);
  UU   :=P2*FRAC(0.2591+1342.2278*T);
  C3[0]:=1.0;          S3[0]:=0.0;
  C3[1]:=COS(M3);      S3[1]:=SIN(M3);      C3[-1]:=C3[1];       S3[-1]:=-S3[1];
  FOR I:=2 TO 7
   DO          ADDTHE(C3[I-1],S3[I-1],C3[1],S3[1],C3[I],S3[I]);
  PERTVEN; PERTMAR; PERTJUP; PERTSAT; PERTMOO;
  DL:=DL + 6.40*SIN(P2*(0.6983+0.0561*T))
         +      1.87*SIN(P2*(0.5764+0.4174*T))
         + 0.27*SIN(P2*(0.4189+0.3306*T))
         +      0.20*SIN(P2*(0.3581+2.4814*T));
  L := 360.0*FRAC(0.7859453 + M3/P2 +
       ((6191.2+1.1*T)*T+DL)/1296.0E3);
  R := 1.0001398  -  0.0000007*T  +  DR*1E-6;
  B :=  DB/3600.0;

 END  (* SUN200  *)
(*-------------------------------------------------------------------------*)
```

Подпрограмма SUN200 показывает, что относительное движение Солнца и Земли не может быть описано с достаточной точностью простой эллиптической орбитой. Наряду с возмущениями со стороны остальных планет (особенно Венеры и Юпитера), в программе учтены месячные колебания центра Земли относительно барицентра системы Земля—Луна. Строго говоря, по эклиптике вокруг Солнца движется не сама Земля, а этот барицентр. Вращение Луны вокруг Земли выражается в небольших периодических возмущениях геоцентрической орбиты Солнца. На подробностях программы и ее алгоритме мы здесь не будем останавливаться.

Все это будет детально рассмотрено в гл.5, где представлены соответствующие программы для всех планет.

2.6. Программа КОКО

Все описанные до сих пор подпрограммы собраны здесь в одну полную программу. Ниже приведены только программы ввода и вывода и основная программа. Уже известные подпрограммы следует вставить в указанные места.

```
(*-----------------------------------------------------------------*)
(*                            KOKO                                 *)
(*                 Преобразование координат                        *)
(*                      Версия 28 12.88                            *)
(*-----------------------------------------------------------------*)
PROGRAM KOKO(INPUT,OUTPUT);

  TYPE REAL33 = ARRAY[1 .3,1. 3] OF REAL,

  VAR X,Y,Z,XS,YS,ZS:  REAL,
      T,TEQX,TEQXN .   REAL,
      LS,BS,RS       : REAL;
      A              : REAL33;
      ECLIPT         : BOOLEAN;
      MODE           : CHAR;
(*-----------------------------------------------------------------*)
(* Сюда вводятся подпрограммы в такой последовательности:          *)
(*     SN, SC, ATN, ATN2, CART, POLAR, GGG, GMS, MJD, ECLEQU       *)
(*     EQUECL, PMATECL, PMATEQU, PRECART, SUN200                   *)
(*-----------------------------------------------------------------*)
PROCEDURE GETEQX(VAR TEQX.REAL);
  BEGIN
    WRITE  (' равноденствие (YYYY) ?     ');
    READLN (TEQX); TEQX := (TEQX-2000)/100;
 END;
(*-----------------------------------------------------------------*)
PROCEDURE GETDAT(VAR T:REAL);
 VAR D,M,Y   :  INTEGER;
      HOUR,JD: REAL;
   BEGIN
    WRITE (' дата (день месяц год час) ?           ');
    READLN (D,M,Y,HOUR),
    JD  :=  MJD(D,M,Y,HOUR) + 2400000.5;
    WRITELN; WRITELN (' JD',JD:13:4), WRITELN;
    T   := (JD-2451545.0)  /  36525.0,
  END;
(*-----------------------------------------------------------------
```

```
PROCEDURE GETINP (VAR X,Y,Z,TEQX.REAL; VAR ECLIPT: BOOLEAN);
  VAR I,G,M  :  INTEGER;
      L,B,R,S:  REAL;
  BEGIN (* GETINP  *)

    WRITELN;
    WRITELN('       KOKO: Преобразование координат               ');
    WRITELN('                 Версия 28 12.88                    ');
    WRITELN('      (c) 1988 Thomas Pfleger, Oliver Montenbruck   ');
    WRITELN;
    WRITELN ('Ввод координат· выберите, пожалуйста, необходимое  ');
    WRITELN;
    WRITELN ('1  эклипт. прямоугольные   2  эклипт.   полярные   ');
    WRITELN ('3  экватор. прямоугольные  4  экватор.  полярные   ');
    WRITELN;
    WRITE  ('   '); READLN (I); WRITELN;
    CASE I OF
      1: BEGIN
           WRITE('  координаты (x,y,z) ?         ');
           READLN(X,Y,Z);  ECLIPT =TRUE;
         END;
      2: BEGIN
         WRITE (' координаты (L (o '' "), B (o '' "), R) ?           ');
         READ(G,M,S); GGG(G,M,S,L); READLN(G,M,S,R); GGG(G,M,S,B);
         CART(R,B,L,X,Y,Z);    ECLIPT:=TRUE;
         END;
      3: BEGIN
           WRITE (' координаты (x,y,z) ?         ');
           READLN(X,Y,Z);  ECLIPT:=FALSE;
          END;
      4: BEGIN
           WRITE(' координаты (Ra (h m s), Dec (o '' "), R) ?        ');
           READ(G,M,S);  GGG(G,M,S,L),
           READLN(G,M,S,R); GGG(G,M,S,B);
           L:=L*15.0;   CART(R,B,L,X,Y,Z);   ECLIPT:=FALSE;
        END;
     END;            (* CASE                     *)
   GETEQX (TEQX);(* ввод равноденствия           *)

 END;                (* GETINP                   *)
(*--------------------------------------------------------------*)
PROCEDURE RESULT (X,Y,Z: REAL; ECLIPT: BOOLEAN);
  VAR L,B,R,S:  REAL;
      G,M    :  INTEGER;
  BEGIN              (* RESULT                   *)
    WRITELN;
```

```
    WRITELN (' (x,y,z) = (',x:13:8,',',y.13:8,',',z:13:8,'          );
    WRITELN;
    POLAR  (X,Y,Z,R,B,L);
    IF ECLIPT
     THEN
      BEGIN
        WRITELN (' ':5,'  o  "  " ','"  .8,'  o "  "        ');
        GMS(L,G,M,S); WRITE(' L = ',G 3,M·3,S:5:1,'':3);
        GMS(B,G,M,S); WRITE(' B = ',G 3,M:3,S·5:1,'':3);
      END
    ELSE
      BEGIN
         WRITELN (' ':5,'  h m s  ','':10,'   o "  " ' );
         GMS(L/15,G,M,S); WRITE(' RA  = ',G·2,M.3,S:5:1,'      ':3);
         GMS(B,G,M,S),    WRITE(' DEC = ',G:3,M:3,S:5:1,'      ':3);
      END;
   WRITELN (' R = ',R·12·8); WRITELN;
 END;      (* RESULT    *)
(*------------------------------------------------------------*)
BEGIN (* KOKO     *)
   GETINP (X,Y,Z,TEQX,ECLIPT);
   RESULT (X,Y,Z,ECLIPT);
   REPEAT
    WRITE (' команда (? = подсказка) '); READLN (MODE); WRITELN;
    IF   MODE   IN   ['?','P','p','E','e','A','a','H','h','G','g']   THEN
    CASE MODE OF
 '?'  : BEGIN(* подсказка   *)
          WRITELN,
          WRITELN('  A  -> экваториальный E: -> эклиптический    ');
          WRITELN('  P. -> прецессия      G: -> геоцентрический  ');
          WRITELN('  H: -> гелиоцентрический  S: -> STOP          ');
          WRITELN,
        END,
 'P','p': BEGIN      (* Прецессия       *)
             WRITE('   новое'),
             GETEQX(TEQXN),(* ввод нового равноденствия    *)
             IF ECLIPT THEN PMATECL(TEQX,TEQXN,A)
                      ELSE PMATEQU(TEQX,TEQXN,A);
             PRECART(A,X,Y,Z),
             TEQX  = TEQXN,
             WRITELN,
             WRITELN(' координаты равноденствия T  =',TEQX:13:10);
           END;
 'E','e': BEGIN       (* -> эклиптические     *)
            WRITELN;
            IF NOT ECLIPT THEN EQUECL (TEQX,X,Y,Z);
```

```
              ECLIPT = TRUE,
              WRITELN(' эклиптические координаты.          ');
            END,
 'A','a': BEGIN        (* -> экваториальные                  *)
              WRITELN,
              IF ECLIPT THEN ECLEQU (TEQX,X,Y,Z);
              ECLIPT:= FALSE,
              WRITELN(' экваториальные координаты'       ');
            END,
 'G','g',     (* -> геоцентрические координаты      *)
 'H','h':   (* -> гелиоцентрические координаты      *)
              BEGIN
               GETDAT(T), (* ввод дат    *)
               SUN200(T,LS,BS,RS),
               CART(RS,BS,LS,XS,YS,ZS),
               PMATECL(T,TEQX,A),
               PRECART(A,XS,YS,ZS),
               IF NOT ECLIPT THEN ECLEQU(TEQX,XS,YS,ZS);
               IF  MODE  IN  ['G','g']
                   THEN (* -> геоцентрич              *)
                    BEGIN
                     X:=X+XS,   Y =Y+YS,   Z =Z+ZS,
                     WRITELN(' геоцентрические координаты:    ');
                     END
                  ELSE (* -> гелиоцентр               *)
                    BEGIN
                     X =X-XS,   Y =Y-YS,   Z =Z-ZS,
                     WRITELN(' гелиоцентрические координаты:   ');
                   END,
               END,
          END,
    IF  NOT  (MODE  IN  ['?','S','s'])  THEN  RESULT(X,Y,Z,ECLIPT);
 UNTIL  MODE  IN  ['S','s'],
END. (* KOKO     *)
(*-----------------------------------------------------------    *)
```

Используя простое меню, можно вызвать отдельные функции программы. После ввода эклиптических или экваториальных координат точки с помощью различных команд можно перейти к любой другой системе координат:

A преобразование в экваториальные координаты,
E преобразование в эклиптические координаты,
P прецессия (изменение равноденствия),
G преобразование в геоцентрические координаты,
H преобразование в гелиоцентрические координаты,
S остановка (конец программы)

Далее приводится наглядный пример эксплуатации и возможностей программы КОКО. Пересчитываемые координаты могут быть даны в экваториальной или эклиптической системе При этом можно выбирать, представлять ли их в прямоугольных или полярных координатах. Этот выбор и запрашивает КОКО для первого ввода (все вводимые данные приведены курсивом).

КОКО: Преобразование координат
Версия 28.12.88
(с) 1988 Thomas Pfleger, Oliver Montebruck

Ввод координат: выберите, пожалуйста, необходимое
1 эклипт. прямоугольные 2 эклипт. полярные
3 экватор. прямоугольные 4 экватор. полярные
4

Мы выбрали экваториальные координаты в полярном представлении и задаем координаты точки весны для равноденствия 1950. Расстояние произвольно выбрано равным 1 а.е. Оно должно быть положительным и отличаться от нуля. КОКО подтверждает ввод, выводя эти данные в прямоугольных и полярных координатах

Координаты (Ra (h m s), Dec (° ′ ″), R) ? *0 0 0.0 0 0 0.0 1.0*
равноденствие (YYYY) ? *1950 0*

(x,y,z) = (1.00000001, 0.00000000, 0.00000000)

RA = $0^h\ 0^m\ 0^s.0$ DEC = $0^\circ\ 0'\ 0''.0$ R = 1.00000001

В первую очередь мы хотим данное положение пересчитать на равноденствие 2000 и выбираем вариант Р для вычисления прецессии. После ввода года мы получим положение точки равноденствия на момент 2000. Прецессия составит при этом более половины градуса.

Команда (? = подсказка): *P*
новое равноденствие (YYYY) ? *2000 0*
координаты равноденствия T = 0.0000000000
(x, y, z) = (0.99992572, 0 01117889, 0.00485898)

RA = $0^h\ 2^m\ 33^s.7$ DEC = $0^\circ\ 16'\ 42''.2$ R = 1.000000002

Теперь заданные экваториальные координаты нужно пересчитать в эклиптическую систему. Для этого выбираем вариант Е и получаем пересчитываемые значения опять в виде прямоугольных и полярных координат. На месте прямого восхождения и склонения здесь будут стоять долгота L и широта B Расстояние, как и в предыдущем примере, остается неизменным.

Команда (? = подсказка): *E*
эклиптические координаты:
(x, y, z) = (0.99992572, 0.01218922, 0.00001132)

$L = 0^\circ\ 41'\ 54''.3 \qquad B = 0^\circ\ 0'\ 2''3 \qquad R = 1.00000002$

Используя вариант H, теперь можно перевести данные координаты из геоцентрической системы в гелиоцентрическую. Однако КОКО не позволяет проверить, к какой из этих двух систем относятся действительные данные. Можно многократно один за другим вызывать варианты H и G, получая при этом бессмысленные результаты. Потребитель сам в этом случае должен контролировать процесс счета. Выбирая преобразование «геоцентрические—гелиоцентрические» (H) или обратное ему (G), нужно задать дату и время. В ответ КОКО выдает соответствующую юлианскую дату и данные, переведенные в нужную систему. Поскольку в предыдущем диалоге были выбраны эклиптические координаты, теперь мы получаем гелиоцентрические эклиптические координаты. Нужно учесть, что расстояние при этом меняется

Команда (? = подсказка). *H*
дата (день месяц год час) ? *1 1 1989 0.0*
JD 2447527.5000
гелиоцентрические координаты·

$(x, y, z) = (\ 0.81725250,\ 0.97838164,\ 0.00003596)$

$L = 50^\circ\ 7'\ 39''5 \qquad B = 0^\circ\ 0'\ 5''.8 \qquad R = 1.27480677$

Пересчет обратно в эклиптические координаты может быть сделан с помощью варианта A, который преобразует текущие (эклиптические) координаты в экваториальные.

Команда (? = подсказка). *A*
экваториальные координаты·

$(x, y, z) = (\ 0.81725250,\ 0.89763330,\ 0.38921086)$

$RA = 3^h\ 10^m\ 44^s1 \qquad DEC = 17^\circ\ 46'\ 36''.5 \qquad R = 1.27480677$

Теперь приведем текущие координаты опять к равноденствию 1950.

Команда (? = подсказка): *P*
новое равноденствие (YYYY) ? *1950.0*
координаты равноденствия T = -0.5000000000
$(x, y, z) = (\ 0.82911751,\ 0.88843067,\ 0.38521087)$

$RA = 3^h\ 7^m\ 54^s7 \qquad DEC = 17^\circ\ 35'\ 17''.2 \qquad R = 1.27480678$

Преобразование этих значений в геоцентрические координаты должно дать исходные координаты точки равноденствия, так как после первых трех преобразований опять вызывается обратное преобразование. Это предполагает, что задаются те же значения времени, которые были использованы при вычислении гелиоцентрических координат. Однако неизбежные ошибки округления при счете почти всегда не позволяют получить точно те же координаты, которые были заданы вначале.

Команда (? = подсказка): *G*
дата (день месяц год час) ? *1 1 1989 0 0*
JD 2447527 5000
геоцентрические координаты·
(x, y, z) = (1 00000002, 0 00000001, 0.00000000)
RA = $0^h\ 0^m\ 0^s0$ DEC = $0^\circ\ 0'0''0$ R = 1 00000002

Ввод S заканчивает работу программы.
Команда (? = подсказка) *S*

В зависимости от используемого компилятора Паскаля полученные значения могут немного отличаться от приведенных выше.

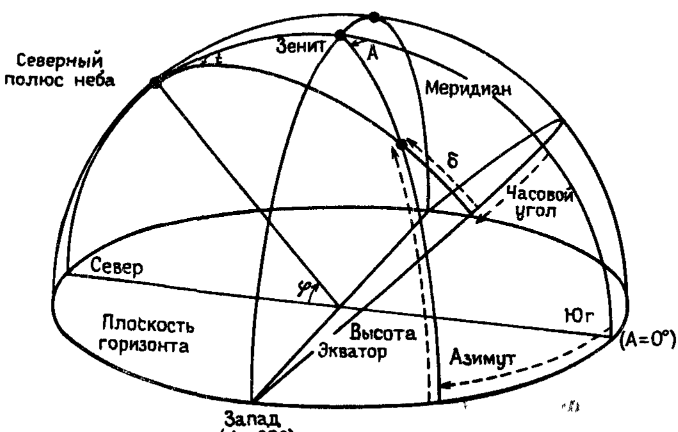
3. РАСЧЕТ ВОСХОДОВ И ЗАХОДОВ

3.1. Горизонтальная система координат наблюдателя

В рассматривавшихся до сих пор эклиптической и экваториальной системах координат особую роль играют средняя плоскость орбиты Земли и положение земной оси Однако для наблюдателя на поверхности Земли обе эти фиксированные в пространстве системы непосредственно не применимы Так как наблюдатель принимает участие в суточном вращении Земли, не замечая этого, у него создается впечатление, что в течение суток Солнце, Луна и звезды движутся по большим дугам с востока на запад, достигая своего наивысшего положения над горизонтом в меридиане

Ход видимых траекторий звезд зависит от географической широты места наблюдения В Южном полушарии Земли большинство светил достигает наибольшей высоты над горизонтом не на юге, а на севере Для наблюдателя, привыкшего к виду неба умеренных северных широт, на юге, естественно, ориентация затрудняется. Известные созвездия кажутся ему перевернутыми.

На небесной сфере особо выделены две точки· зенит (точка, расположенная вертикально над наблюдателем) и Северный полюс мира (рис 3 1) Это точка видимой небесной сферы, указывающая направление земной оси, наблюдателю кажется, что все светила вращаются вокруг этой точки по концентрическим кругам, причем ее высота над го-



ис.3.1. Горизонтальная система.

ризонтом соответствует географической широте ϕ наблюдателя. Большой круг, проходящий через зенит и Северный полюс мира (его называют меридианом), делит горизонт как раз в направлении север—юг.

Координатами в горизонтальной системе являются азимут A и высота h, которые могут быть определены, например, при помощи теодолита (см. рис.3.1). Высота над горизонтом—это угол, на который надо отклонить теодолит вверх от горизонтального положения. Азимут—угол, на который необходимо повернуть теодолит вокруг вертикальной оси, чтобы направить его на искомую точку, начиная от направления на юг. Согласно такому определению, азимут точки запада $A = 90°$, а азимут точки востока $A = 270°$ (или $A = -90°$). К сожалению, наряду с этим определением употребляется и другое, принятое в навигации. Там азимут отсчитывают от точки севера. При этом точка востока имеет азимут $A = 90°$. В случае сомнения всегда нужно проверять, какое именно определение использовалось!

Вращение Земли приводит к постоянному изменению азимута и высоты светила с заданным прямым восхождением α и склонением δ. Если зафиксировать положение телескопа в горизонтальной системе, то через поле зрения телескопа будут проходить звезды с одинаковыми склонениями, но с различными прямыми восхождениями. Поэтому определение склонения светила по азимуту и высоте не зависит от времени. Однако прямое восхождение можно определить только относительно звезд, находящихся в меридиане, как разность двух значений.

Для взаимного преобразования горизонтальных и экваториальных координат было введено понятие часового угла τ. Часовой угол—это разность между прямым восхождением наблюдаемой звезды и прямым восхождением звезды в меридиане. Как и прямое восхождение, часовой угол обычно измеряется в единицах времени ($1^h \equiv 15°$) и приблизительно[1] равен времени, прошедшему после последнего прохождения звезды через меридиан. При определениях, принятых нами, A и τ имеют одинаковый знак. Для прямоугольных координат

$$
\begin{aligned}
x &= \cos h \cos A, &\qquad \hat{x} &= \cos\delta \cos\tau,\\
y &= \cos h \sin A, &\qquad \hat{y} &= \cos\delta \sin\tau,\\
z &= \sin h, &\qquad \hat{z} &= \sin\delta
\end{aligned}
\tag{3.1}
$$

Справедливы преобразования

$$
\begin{aligned}
x &= +\hat{x}\sin\varphi - \hat{z}\cos\varphi, &\qquad \hat{x} &= +x\sin\varphi + z\cos\varphi,\\
y &= +\hat{y}, &\qquad \hat{y} &= +y,\\
z &= +\hat{x}\cos\varphi + \hat{z}\sin\varphi, &\qquad \hat{z} &= -x\cos\varphi + z\sin\varphi.
\end{aligned}
\tag{3.2}
$$

[1] Один оборот Земли вокруг своей оси происходит за 23^h56^m.

Здесь φ — географическая широта наблюдателя, а знак «^» обозначает экваториальные координаты.

Для определения моментов восхода и захода светил, что является темой этой главы, потребуется только одно равенство:

$$z = \hat{x} \cos \varphi + \hat{z} \sin \varphi.$$

После преобразований его можно переписать в виде

$$\sin h = \cos \varphi \cos \delta \cos \tau + \sin \varphi \sin \delta, \tag{3.3}$$

что позволяет вычислить высоту h по данным значениям географической широты φ, склонения δ и часового угла τ. Прежде чем мы воспользуемся этими соотношениями, нужно проделать некоторую предварительную работу. В первую очередь отметим, что мы не умеем определять координаты (α, δ) Солнца и Луны. Кроме того, мы не знаем, как в заданный момент времени по известному прямому восхождению вычислить часовой угол. Наконец, мы должны разобраться с целым рядом поправок, влияющих на наблюдаемую высоту горизонта. Только после этого мы вернемся к приведенному выше уравнению.

3.2. Солнце и Луна

При вычислении моментов восхода и захода не требуется высокая точность координат Солнца и Луны. Поэтому подпрограммы MINI_SUN и MINI_MOON для описания соответствующей орбиты содержат только главные члены. Речь идет о сильно сокращенных вариантах подпрограмм SUN200 и MOON, которые будут подробно рассматриваться в гл.5 и 6. Пересчет эклиптических долготы и широты в экваториальные координаты уже заложен в эти подпрограммы, так что они полностью обеспечивают вычисление прямого восхождения и склонения объекта.

```
(*-----------------------------------------------------------*)
(* MINI_MOON: Координаты Луны с малой точностью (около 5'/1') *)
(* T:  время в юлианских столетиях с J2000 (T=(JD-2452545)/36525) *)
(* RA :  прямое восхождение (h)                              *)
(* DEC:  склонение (град.)                                   *)
(*       (равноденствие даты)                                *)
(*-----------------------------------------------------------*)
PROCEDURE MINI_MOON (T: REAL; VAR RA,DEC: REAL):
  CONST P2= 6.283185307;     ARC = 206264.8062;
    COSEPS=0.91748; SINEPS=0.39778; (*cos/sin (наклон эклипт.) *)
  VAR   L0,L,LS,F,D,H,S,N,DL,CB    : REAL;
        L_MOON,B_MOON,V,W,X,Y,Z,RHO:REAL;
  FUNCTION FRAC(X:REAL):REAL;
  (* вместо TRUNC при T<-24 использовать LONG_TRUNC или INT !  *)
  BEGIN X:=X-TRUNC(X); IF (X<0) THEN X:=X+1; FRAC:=X END;
  BEGIN
```

```
(* средние элементы орбиты Луны   *)
L0:=FRAC(0 606433+1336 855225*T);      (* ср.долгота Луны (рад.)      *)
L :=P2*FRAC(0.374897+1325 552410*T);    (* ср. аномалия Луны           *)
LS:=P2*FRAC(0.993133+   99.997361*T);   (* ср. аномалия Солнца         *)
D :=P2*FRAC(0.827361+1236.853086*T);    (*  разн.долгот  Луна—Солнце*)
F :=P2*FRAC(0 259086+1342.227825*T);    (* восходящий узел             *)
     DL:= +22640*SIN(L) - 4586*SIN(L-2*D) + 2370*SIN(2*D)+
              769*SIN(2*L)     - 668*SIN(LS) - 412*SIN(2*F)-
              212*SIN(2*L-2*D)  -  206*SIN(L+LS-2*D)  +192*SIN(L+2*D)-
              165*SIN(LS-2*D)  -  125*SIN(D)  -  110*SIN(L+LS)+
              148*SIN(L-LS)   -   55*SIN(2*F-2*D);
     S := F  +  (DL+412*SIN(2*F)+541*SIN(LS))/ARC;
     H := F-2*D,
     N := -526*SIN(H) + 44*SIN(L+H) - 31*SIN(-L+H) - 23*SIN(LS+H)
         + 11*SIN(-LS+H)  -25*SIN(-2*L+F)  +  21*SIN(-L+F);
     L_MOON:= P2 * FRAC ( L0 + DL/1296E3 ); (* радианы   *)
     B_MOON:= ( 18520 0*SIN(S) + N ) / ARC;   (* радианы   *)
     (* экваториальные координаты   *)
     CB . =COS(B_MOON);
     X  : =CB*COS(L_MOON),
     V  : =CB*SIN(L_MOON),
     W  : =SIN(B_MOON);
     Y  : =COSEPS*V-SINEPS*W;
     Z  : =SINEPS*V+COSEPS*W;
     RHO. =SQRT(1.0-Z*Z),
     DEC: =(360.0/P2)*ARCTAN(Z/RHO);
     RA : =( 48.0/P2)*ARCTAN(Y/(X+RHO));
     IF RA<0 THEN RA:=RA+24 0;
  END;
(*-----------------------------------------------------------------*)
(* MINI_SUN : Координаты Солнца с малой точностью  (около 1')      *)
(*        T : время в юлианских  столетиях с J2000                 *)
(*            (T=(JD-2452545)/36525)                               *)
(*       RA : прямое восхождение (h)                               *)
(*      DEC: склонение (град.)                                     *)
(*            (равноденствие даты)                                 *)
(*-----------------------------------------------------------------*)
PROCEDURE MINI_SUN(T:REAL; VAR RA,DEC: REAL);
  CONST P2=6.283185307; COSEPS=0.91748; SINEPS=0.39778;
  VAR L,M,DL,SL,X,Y,Z,RHO: REAL;
  FUNCTION FRAC(X:REAL):REAL;
   BEGIN X:=X-TRUNC(X); IF (X<0) THEN X:=X+1; FRAC:=X  END;
  BEGIN
    M  := P2*FRAC(0.993133+99.997361*T);
    DL := 6893.0*SIN(M)+72.0*SIN(2*M);
    L  := P2*FRAC(0.7859453 + M/P2 + (6191.2*T+DL)/1296E3);
```

```
    SL := SIN(L);
    X  := COS(L); Y:=COSEPS*SL; Z:=SINEPS*SL; RHO:=SQRT(1.0-Z*Z);
    DEC:= (360.0/P2)*ARCTAN(Z/RHO);
    RA := ( 48.0/P2)*ARCTAN(Y/(X+RHO));
    IF (RA<0) THEN RA:=RA+24.0;
  END;
(*-----------------------------------------------------------------*)
```

3.3. Звездное время и часовой угол

Чтобы вычислить, на какой высоте находятся Солнце и Луна над горизонтом, недостаточно знать их прямое восхождение. Мы должны знать также прямое восхождение звезд, проходящих через меридиан, и по нему определять часовой угол в зависимости от места и времени наблюдения. В процессе суточного вращения Земли наблюдатель видит, как через меридиан проходят звезды с прямыми восхождениями от 0^h до 24^h В течение одного часа прямое восхождение кульминирующих звезд меняется на 1^h

Регулярное движение звезд дает возможность **ввести** новое исчисление времени, называемое *звездным временем* **Для каждого** места оно определяется как прямое восхождение той звезды, которая находится точно в меридиане. В соответствии с этим определением наблюдатель сам может определить звездное время

Необходимость введения звездного времени в дополнение к солнечному возникла из-за небольшого различия между продолжительностью солнечных суток и периодом обращения Земли вокруг своей оси. Часы, которыми мы обычно пользуемся, калиброваны таким образом, что сутки поделены на 24^h. Движение Солнца определяет сутки как смену дня и ночи. Интервал 24^h—это как раз среднее время между двумя прохождениями Солнца через меридиан. Если определить соответствующий промежуток времени для любой звезды, то он будет равен $23^h\,56^m\,4{,}091^s$. Этот более короткий временной интервал, который в отличие от солнечных суток называется звездными сутками, равен времени обращения Земли вокруг своей оси. Причиной различия примерно на 4^m является годичное движение Земли вокруг Солнца. В результате этого движения прямое восхождение Солнца изменяется за год на $360° = 24^h$, а за [сут]ки примерно на 4^m Поэтому промежуток времени между двумя про[хож]дениями через меридиан для Солнца длиннее, чем для звезд, как [раз] на эту величину.

Для данного момента всемирного времени UT можно вычислить звезд[ное] время для Гринвича:

$$= 24110.54841^s + 8640184.812866^s\,T_0 + 1.0027379093\ UT +$$

$$+ 0.093104^s\,T^2 - 0.0000062^s\,T^3, \qquad (3.4)$$

$$T_0 = \frac{JD_0 - 2451545}{36525} \quad \text{и} \quad T = \frac{JD - 2451545}{36525}.$$

Здесь JD и JD_0 — юлианские даты соответственно момента наблюдения и 0^h всемирного времени в дату наблюдения Для местности с географической долготой λ звездное время отличается от гринвичского звездного времени на $\lambda/15°$·

$$\theta = \theta_0 - \lambda \cdot 1^h/15°. \tag{3.5}$$

В этом выражении *западная долгота считается положительной* (для Мюнхена, например, $\lambda = -11{,}6°$) Для звезды с прямым восхождением α часовой угол τ будет равен

$$\tau = \theta - \alpha. \tag{3.6}$$

```
(*-----------------------------------------------------------------*)
(* LMST: среднее местное звездное время                            *)
(*-----------------------------------------------------------------*)
FUNCTION LMST(MJD,LAMBDA.REAL):REAL;
  VAR MJD0,T,UT,GMST            : REAL;
  FUNCTION FRAC(X:REAL)         : REAL;
    BEGIN X:=X-TRUNC(X); IF (X<0) THEN X:=X+1; FRAC:=X  END;
  BEGIN
   MJD0  : =TRUNC(MJD);              (* Standard Pascal        *)
   (*  MJD0· =INT(MJD);          *)  (* TURBO Pascal           *)
   (*  MJD0: =LONG_TRUNC(MJD);*)     (* ST Pascal plus         *)
   UT    . =(MJD-MJD0)*24,     T =(MJD0-51544 5)/36525.0;
   GMST  : =6 697374558  +  1 0027379093*UT
            +(8640184.812866+(0 093104-6 2E-6*T)*T)*T/3600.0;
   LMST  : =24.0*FRAC( (GMST-LAMBDA/15 0) / 24 0 );
 END;
(*-----------------------------------------------------------------*)
```

3.4. Всемирное и эфемеридное время

Мы до сих пор употребляли понятие «время», особенно не задумываясь о его смысле Но в астрономии параллельно используют целый ряд временных шкал, которые можно в основном разделить на два класса. Среди них наиболее важны *динамическое время* (TDB/TDT) и *всемирное время* (UT). Рассмотрим немного подробнее, какие задачи лежат в основе этих систем счета времени.

По своему замыслу динамическое время — это время, лежащее в основе описания астрономических явлений на языке физики. Секунда динамического времени определяется как фиксированное число периодов определенного перехода в атоме цезия, и сегодня она измеряется при помощи атомных часов В 1984 г динамическим временем было заменено *эфемеридное время* (ET) Определение эфемеридного времени основывалось на астрономических эфемеридах, т е таблицах движения Солнца, Луны и планет, вычисленных по законам классической механики Эфемеридное время можно было определить сравнением наблюдаемых положений небесных тел с заранее вычисленными эфемеридами. Основанием для введения динамического времени послужило положение теории относительности, согласно которому универсального времени не существует, а течение времени зависит от положения и движения системы координат, в которой измеряется время Это, в частности, определяет важное различие между земным (измеряемым относительно Земли) динамическим временем TDT и временем TDB, относимым к центру масс Солнечной системы. Однако при решении наших задач мы можем считать все три системы счета времени—ET, TDB и TDT—эквивалентными.

В противоположность эфемеридному и динамическому времени шкала *всемирного времени* (UT) *неравномерна* UT сегодня является лучшим представлением солнечного времени Оно было введено, чтобы сутки в течение нескольких тысяч лет в среднем равнялись 24^h Правда, при этом продолжительность секунды всемирного времени становится непостоянной, так как фактическая средняя длина суток зависит от вращения Земли и видимого движения Солнца (т.е. от продолжительности года). К сожалению, невозможно определить всемирное время пересчетом динамического времени, так как движение Земли непредсказуемо. Любое изменение скорости вращения Земли меняет и продолжительность суток, что должно быть учтено во всемирном времени. Поэтому всемирное время определяют как функцию звездного времени, которое непосредственно отражает вращение Земли. 0^h всемирного времени каждого дня определяется как момент, когда среднее звездное время в Гринвиче (GMST) принимает значение

$$\mathrm{GMST}\ (0^h\mathrm{UT}) = 24110{,}54841^s + 8640184{,}812866^s T_0 + $$
$$+ 0{,}093104^s T_0^2 - 0{,}0000062^s T_0^3,$$

где

$$T_0 = \frac{\mathrm{JD}(0^h\mathrm{UT}) - 2451545}{36525}.$$

Это равенство уже встречалось в разд 3.3, где мы использовали его при определении звездного времени по заданному всемирному. Но надо иметь в виду, что именно звездное время является наблюдаемой величиной, по которой затем определяют всемирное время

Разность между всемирным и динамическим (или эфемеридным) временем можно определить только для прошлого. В табл.3.1 даны значения величины ΔT = ET - UT (ET = TDB = TDT) в разные годы нашего столетия. Сейчас ΔT увеличивается примерно на 0,5—1 сек в год.

Таблица 3 1 Ход разности ΔT = ET − UT в секундах

год	ET—UT	Год	ET-UT	Год	ET-UT	Год	ET-UT
1900	-2,72	1925	23,62	1950	29,15	1975	45,58
1905	3,86	1930	24,02	1955	31,05	1980	50,24
1910	10,46	1935	23,93	1960	33,15	1985	54,34
1915	17,20	1940	24,33	1965	35,73	1990	56,7
1920	21,16	1945	26,77	1970	40,18		

В повседневной жизни обычно используется *координированное всемирное время* (UTC) UTC измеряется атомными часами и меняется с той же скоростью, что и динамическое время Дополнительные секунды, которые могут быть добавлены два раза в год, позволяют добиться того, что UTC не отличается от всемирного времени UT более, чем на 0,9 сек UTC служит основой для измерения времени по всей Земле Для этого каждая местность отнесена к какому либо часовому поясу Официально принятое в часовом поясе время отличается от UTC на целое (или полуцелое) число часов На практике часовые пояса приспосабливаются к географическим реалиям, прежде всего к государственным границам, чтобы по возможности избежать разделения одной страны на несколько часовых поясов Принадлежность определенной местности к тому или иному поясу устанавливается международным соглашением и наносится на карту часовых поясов

Теперь, когда мы познакомились с разными шкалами времени, возникает естественный вопрос, какое время мы используем при вычислении восходов и заходов Солнца и Луны Из приведенных выше определений следует общее правило для использования всемирного и эфемеридного времени

* Всемирное время используют для вычисления звездного времени
* Эфемеридное (ET) или динамическое время (TDB/TDT) используют для вычисления эфемерид Солнца, Луны и планет

В качестве примера рассмотрим отдельные шаги при вычислении высоты Луны над горизонтом Вычисления проводились для 1 января 1982г , 0^h среднеевропейского времени (CET), место наблюдения—Мюнхен (φ = 48,1°, λ = −11,6°) Незначительное различие между UTC и UT в дальнейших вычислениях игнорируется, так что всемирное время у нас будет равно UT = CET − 1^h Согласно приведенному выше определению, для вычисления звездного времени мы должны использовать всемирное время, а эфемеридное время применяется при вычислении координат Луны Это приводит к следующим преобразованиям, которые представлены в небольшом фрагменте программы

```
CONST DAY=1, MONTH= 1, YEAR=1982, CET=0 0,  (* дата           *)
      UT_MINUS_CET  = -1 0,                 (* поясное время  *)
      ET_MINUS_UT   = 52 17,                (* 1982, секунды  *)
```

```
    PHI=48 1, LAMBDA=-11 6,                    (* геогр координаты       *)
VAR UT,MJD_UT,MJD_ET,T_UT,T_ET   REAL,
    RA,DEC,THETA,TAU,SIN_H       REAL,
BEGIN
  UT     = CET + UT_MINUS_CET,
  MJD_UT = MJD(DAY,MONTH,YEAR,UT),            (* MJD всемир время   *)
  MJD_ET = MJD_UT + (ET_MINUS_UT/86400 0)  (* MJD эфемер время   *)
  T_ET   = (MJD_ET-51544 5)/36525 0,          (* столетия с J2000   *)
  MINI_MOON(T_ET,RA,DEC),                     (* экватор координ    *)
  THETA = LMST(MJD_UT,LAMBDA),              (* местн звездн время, h  *)
  TAU    = 15 0 * (THETA-RA),                 (* часовой угол, h    *)
  SIN_H  =                                    (* синус высоты       *)
  SN(PHI)*SN(DEC) + CS(PHI)*CS(DEC)*CS(TAU),(*  над горизонтом     *)
END
```

С другой стороны, можно задаться вопросом, к какой ошибке приведет игнорирование различия между всемирным и эфемеридным временем Если при вычислении координат Луны упрощенно использовать UT вместо ET, то приведенный выше пример будет иметь следующий вид

```
  UT     = CET + UT_MINUS_CET,
  MJD_UT = MJD(DAY,MONTH,YEAR,UT),          (* MJD всемир  время  *)
  T_UT   = (MJD_UT-51544 5)/36525 0,          (* столетия с J2000   *)
  MINI_MOON(T_UT,RA,DEC),                     (* экватор  координ   *)
  THETA = LMST(MJD_UT,LAMBDA),              (* местн звезд время, h   *)
  TAU    = 15 0 * (THETA-RA),                 (* часовой угол, h    *)
  SIN_H  =                                    (* синус высоты       *)
  SN(PHI)*SN(DEC) + CS(PHI)*CS(DEC)*CS(TAU), (*  над горизонтом    *)
```

Хотя звездное время рассчитано здесь правильно, вычисленные координаты Луны относятся к более раннему моменту времени, отличающемуся от истинного на величину ΔT = ET − UT В настоящее время ΔT равно примерно 1 мин, а за это время Луна перемещается на 30″ Эта ошибка уже меньше, чем при вычислении координат Луны по программе MINI_MOON Время восхода и захода будет ошибочным примерно на 3^s Для Солнца эта величина будет еще меньше Поэтому в программе SUNSET мы можем с чистой совестью пренебречь значением ΔT В дальнейшей работе, например, при вычислении точных положений планет для предсказания покрытий звезд, необходимо четко различать разные системы счета времени Именно поэтому мы довольно подробно остановились на особенностях исчисления времени в астрономии

3.5. Параллакс и рефракция

До сих пор мы приводили прямое восхождение и склонение любого небесного тела в геоцентрических экваториальных координатах, т.е. в системе координат, привязанной к центру Земли Но мы находимся не в центре Земли, а наблюдаем с ее поверхности К какому же различию между вычисленными и реально наблюдаемыми координатами это приводит?

При наблюдении неподвижных звезд геоцентрические и топоцентрические (в координатной системе, нуль пункт которой сдвинут от центра Земли к наблюдателю на поверхности) координаты практически не отличаются друг от друга Расстояние до звезды несравнимо больше, чем расстояние от наблюдателя до центра Земли Поэтому заметное различие получается только в случае наблюдений близлежащих небесных тел, таких, как планеты, Солнце или Луна

Это различие, называемое *параллаксом*, приводит к следующему эффекту объект с поверхности Земли кажется расположенным ниже, чем он был бы виден из центра Земли Топоцентрическая высота h меньше, чем геоцентрическая h' (рис 3 2) Если объект в зените, то параллакс исчезает При уменьшении высоты он заметно растет и достигает максимального значения при $h = 0$ Это значение называется горизонтальным параллаксом и вычисляется по формуле

$$\pi = \arcsin (R_{\oplus}/\Delta), \tag{3 7}$$

где *расстояние наблюдателя* от центра Земли $R_{\oplus} \approx 6378$ км, а Δ—геоцентрическое расстояние до объекта Если в (3 7) подставить расстояние до Солнца, то получим $\pi_{\odot} = 8{,}8''$ С другой стороны, средний горизонтальный параллакс нашего ближайшего соседа—Луны достигает такой величины, которой уже нельзя пренебрегать

$$\pi_{\text{Луны}} = \arcsin \left\{\frac{6378\ \text{км}}{384400\ \text{км}}\right\} \simeq 57'$$

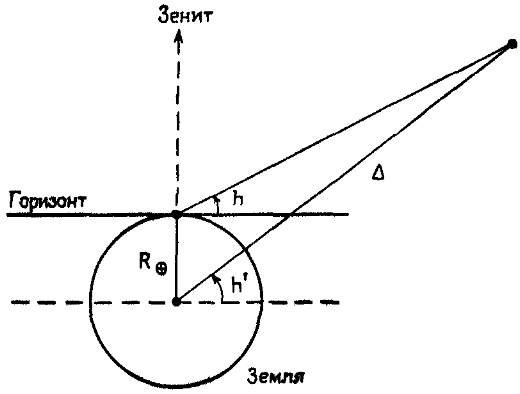


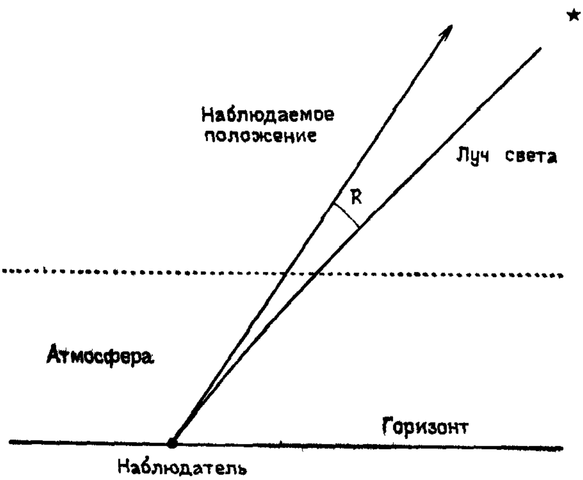
Рис. 3.2. К параллаксу

Теперь понятно, почему для вычисления моментов восхода и захода нужно использовать не геоцентрическую, а топоцентрическую высоту Но прежде чем начать вычисления, нужно рассмотреть еще один эффект

При переходе из космического вакуума в оптически более плотную атмосферу Земли лучи света по закону преломления отклоняются к вертикали (рис 3 3) Это явление называется *атмосферной рефракцией* Из за рефракции наблюдателю на поверхности Земли звезда кажется немного «приподнятой» Луч света, падающий в атмосферу под углом к вертикали, проходит более длинный путь сквозь слои воздуха различной плотности (и коэффициента преломления), чем луч света, падающий вертикально При этом рефракция тем сильнее, чем меньше высота объекта У горизонта рефракция достигает максимальной величины и составляет примерно 34′ В табл 3 2, где приведены некоторые значения рефракции, R—значение, которое нужно добавить к топоцентрической высоте, чтобы получить наблюдаемую высоту Рефракция зависит от давления и температуры воздуха Поэтому приведенные значения могут рассматриваться только как средние Подробные таблицы нормальной рефракции или формулы для численной аппроксимации можно найти в соответствующей литературе

Таблица 3 2 Значения рефракции вблизи горизонта

h	10°	5°	2°	1°	0°
R	5′31″	10′15″	19′7″	25′36″	34′



…с.3.3. К рефракции

Мы уже убедились, что при вычислении моментов восхода и захода нужно исходить из топоцентрической высоты. Одновременно в вычислениях нужно принимать во внимание рефракцию у горизонта. Так как при вычислении моментов восхода и захода имеется в виду восход и заход верхнего края диска Солнца или Луны, то при вычислениях нужно учитывать и их видимый радиус $s_{\odot}$ или $s_{\text{Луны}}$. К моменту восхода или захода светило имеет геоцентрическую высоту

$$h_{\text{в/з}} = 0^{\circ} + \pi - R_{h=0} - s \,. \tag{3.8}$$

Наша задача состоит только в том, чтобы для заранее заданной даты найти момент времени, когда рассматриваемое небесное тело достигнет высоты $h = h_{\text{в/з}}$. Вместо точных значений π и s можно использовать их средние значения, поскольку моменты восхода или захода достаточно вычислять с точностью в несколько минут. Обычно при вычислениях моментов восхода и захода используют следующие значения:

* восход или заход Солнца: $h_{\text{в/з}} = -0^{\circ}50'$,
* восход или заход Луны: $h_{\text{в/з}} = +0^{\circ}08'$,
* восход или заход звезд и планет: $h_{\text{в/з}} = -0^{\circ}34'$.

Поскольку с помощью нашей программы можно также вычислять моменты начала и конца сумерек, приведем еще некоторые определения. Различают:

* *астрономические* сумерки, когда $h = -18^{\circ}$,
* *навигационные* сумерки, когда $h = -12^{\circ}$,
* *гражданские* сумерки, когда $h = -6^{\circ}$.

Эти значения не нужно исправлять за параллакс, рефракцию и видимый радиус. Согласно определению, сумерки наступают или заканчиваются, когда геоцентрическая высота Солнца принимает одно из указанных значений.

Остается решить общую задачу: найти момент, когда светило достигает заданной высоты.

3.6. Восходы и заходы

Равенство (3 3) позволяет вычислить высоту звезды h по заданной географической широте φ, склонению δ и часовому углу τ. При этом часовой угол определяют из (3 6) по известному местному времени.

Напротив, задавая значение высоты h, мы можем преобразовать (3.3) и получим для часового угла следующее выражение:

$$\cos\tau = \frac{\sin h - \sin\varphi\,\sin\delta}{\cos\varphi\,\cos\delta} \,. \tag{3.9}$$

Эта формула дает возможность вычислить часовой угол звезды, когда она наблюдается на высоте h над горизонтом. Равенство определено не для любых значений h, а только для таких высот, которых звезда может реально достичь. Например, при географической широте $\varphi = 50^\circ$ звезда со склонением $\delta = 60^\circ$ будет незаходящей и может наблюдаться только между высотами $h = 20^\circ$ и $h = 80^\circ$ Если значение $\cos \tau$ при заданном значении h больше единицы, то это означает, что звезда никогда не достигает указанной высоты.

При вычислении моментов восхода и захода звезд в формулу (3.9) подставляют значение $h_{в/з} = -0^\circ 34'$ и полученное значение часового угла выражают в единицах времени ($15^\circ \simeq 1^h$). Оно показывает, сколько часов звездного времени нужно восходящей звезде для достижения кульминации Так как в звездных сутках содержится $23^h 56^m 4{,}091^s$ солнечного времени, то коэффициент $23^h 56^m 4{,}091^s / 24^h = 0{,}9972696$ можно использовать для перевода звездного времени в солнечное. Умножив часовой угол на этот коэффициент, получим так называемую половину дневной дуги. Под этим подразумевают половину всего времени видимости звезды

При известном прямом восхождении α_* звезды из уравнения

$$\theta_{в/з} = \begin{cases} \alpha_* - \tau & \text{для восхода,} \\ \alpha_* + \tau & \text{для захода} \end{cases}$$

получим звездное время ее восхода и захода. Обозначим местное звездное время в 0^h через θ_0, тогда звезда будет восходить на $0{,}9973*(\theta_0 - \theta_в)$ часов раньше полуночи и заходить на $0{,}9973*(\theta_з - \theta_0)$ часов позже полуночи.

За сутки Солнце и Луна в отличие от звезд заметно меняют свои координаты. Для вычисления часового угла и моментов восхода и захода светила с помощью приведенных выше уравнений необходимо знать его прямое восхождение и склонение в момент пересечения горизонта. Для этого следует применять метод итерации. Сначала моменты восхода и захода вычисляют, используя координаты для любого момента данных суток (чаще всего выбирают $t = 12^h$). Полученное таким образом приближение позволяет уточнить координаты и соответственно более точно определить моменты восхода и захода. Для Луны эта процедура повторяется до тех пор, пока полученные моменты не будут различаться менее чем на минуту Для Солнца и планет уже первая итерация дает достаточно точные результаты

К сожалению, при определении моментов восхода и захода Луны возникает ряд проблем, которые не могут быть решены применением метода итерации. Мы перечислим некоторые из них:

* Луна, обращаясь вокруг Земли, перемещается по небу с запада на восток, поэтому в отличие от звезд с тем же склонением, она дольше видна на небе. В частности, восход Луны каждые сутки наступает в среднем на 50^m позже, и потому в каждом месяце существуют одни сутки, когда Луна не восходит и одни сутки, ког-

да она не заходит. Например, 8 февраля 1988 г. в Мюнхене Луна восходила в 23^h30^m, а на следующий день зашла в $9^h\ 41^m$. В своем движении относительно звезд, Луна до полуночи оставалась ниже горизонта. Следующий восход Луны был уже не 9 февраля, а 10 февраля в 0^h43^m.

* На высоких географических широтах Солнце и Луна часто целыми сутками остаются над или под горизонтом. Моменты восхода и захода в этом случае определяются в первую очередь суточными изменениями склонения. Методом итерации трудно учесть возможные касания горизонта.

Ввиду этого программа SUNSET основана на иных принципах. Моменты восходов и заходов в ней определяются обратной интерполяцией ряда высот Солнца и Луны. Это требует больших затрат машинного времени, но ведет к существенному упрощению структуры программы.

3.7. Квадратическая интерполяция

Составив таблицу высот, вычисленных с интервалом в 1 ч, можно с помощью простой интерполяции представить ход высоты как функцию времени. Для решения нашей задачи самой подходящей является квадратическая интерполяция: вычисление коэффициентов параболы, аппроксимирующей ход функции между опорными точками, по трем значениям функции. При этом исходят из значений

$$y_- = f(x = -1), \quad y_0 = f(x = 0) \quad \text{и} \quad y_+ = f(x = +1).$$

Затем ищут коэффициенты a, b и c параболы, проходящей через точки $(-1,\ y_-)$, $(0,\ y_0)$ и $(1,\ y_+)$. Подставив их в уравнение параболы, получим три уравнения:

$$y_- = a - b + c,$$

$$y_0 = c,$$

$$y_+ = a + b + c\ ,$$

которые и определяют искомые коэффициенты параболы:

$$\begin{aligned} a &= (y_+ + y_-)/2 - y_0, \\ b &= (y_+ - y_-)/2, \\ c &= y_0. \end{aligned} \tag{3.11}$$

Найденная таким путем парабола при $b^2 \geq 4ac$ имеет нули в точках

$$x_{1,2} = \frac{-b \mp \sqrt{b^2 - 4ac}}{2a} \tag{3.12}$$

и экстремум

$$x_E = \frac{-b}{2a},$$
$$y_E = a\,x_E^2 + b\,x_E + c. \qquad (3.13)$$

Подпрограмма QUAD решает эти уравнения и делает выборку нулей, лежащих между $x = -1$ и $x = +1$.

```
(*------------------------------------------------------------------*)
(* QUAD: вычисление параболы по трем точкам                          *)
(*       (-1,Y_MINUS), (0,Y_0) и (1,Y_PLUS),                         *)
(*       не лежащим на одной прямой.                                 *)
(*                                                                   *)
(*       Y_MINUS,Y_0,Y_PLUS: три значения y                          *)
(*       XE,YE . x и y-значения экстремума параболы                  *)
(*       ZERO1 : первый нуль на отрезке [-1,+1] (для NZ=1,2)         *)
(*       ZERO2 : второй нуль на отрезке [-1,+1] (для NZ=2)           *)
(*       NZ    : число нулей параболы на отрезке [-1,+1]             *)
(*------------------------------------------------------------------*)
PROCEDURE QUAD(Y_MINUS,Y_0,Y_PLUS: REAL,
               VAR XE,YE,ZERO1,ZERO2: REAL; VAR NZ: INTEGER);
  VAR A,B,C,DIS,DX: REAL;
  BEGIN
    NZ := 0;
    A  := 0.5*(Y_MINUS+Y_PLUS)-Y_0, B := 0.5*(Y_PLUS-Y_MINUS);
    C  := Y_0;
    XE := -B/(2.0*A); YE := (A*XE + B) * XE + C;
    DIS:= B*B - 4.0*A*C;  (* дискриминант y = axx+bx+c             *)
    IF (DIS >= 0) THEN    (* парабола пересекает ось x             *)
      BEGIN
        DX := 0.5*SQRT(DIS)/ABS(A); ZERO1:= XE-DX; ZERO2 := XE+DX;
          IF (ABS(ZERO1) <= +1.0) THEN NZ:= NZ + 1;
          IF (ABS(ZERO2) <= +1.0) THEN NZ.= NZ + 1;
          IF (ZERO1<-1.0) THEN ZERO1    := ZERO2;
        END;
    END;
(*------------------------------------------------------------------*)
```

3.8. Программа SUNSET

Программа SUNSET позволяет вычислять моменты восхода и захода Солнца и Луны, а также время начала и конца навигационных сумерек в интервале времени 10 сут. При этом нужно задавать начальную дату 10-суточного промежутка, географические координаты места наблюдения и разность между поясным и всемирным временем.

На рис.3.4 представлена схема поиска времени различных явлений. С помощью подпрограммы SIN_ALT вычисляется синус высоты Солнца и Луны. Высота при этом задается с интервалом в 1 ч. Эти значения интерполируются в QUAD и исследуются на наличие среди них нулевых значений. Если нуль имеется, а Солнце или Луна в начале суток нахо-

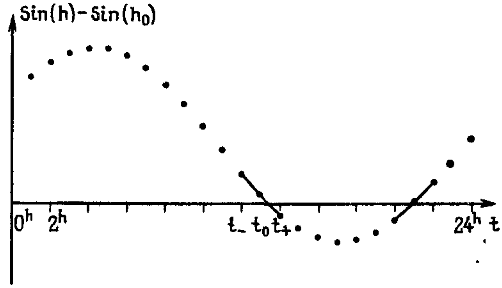


Рис.3.4. Определение моментов восхода и захода.

дятся под горизонтом, получаем момент восхода В противном случае находим момент захода. И наконец, может случиться, что в исследуемом интервале будет два нулевых значения. Чтобы решить, какая из двух точек соответствует восходу, а какая заходу, следует выяснить, где лежит экстремум высоты—над или под горизонтом. Для этого нужно проверять все значения до конца исследуемых суток. Кроме того, нужно определить, не является ли данное светило околополярным и не находится ли оно все время под горизонтом.

Время начала навигационных сумерек определяется таким же способом. Здесь постоянное значение sin(−12°) вычитается из синуса высоты над горизонтом. Если нужно определить момент начала гражданских или астрономических сумерек, то нужно соответственно изменить значение SINH0 [3].

```
(*-----------------------------------------------------------------*)
(*                          SUNSET                                 *)
(*           Моменты восхода и захода Солнца и Луны                *)
(*                     Версия 29.12.1988                           *)
(*-----------------------------------------------------------------*)
PROGRAM SUNSET(INPUT,OUTPUT);
  VAR ABOVE,RISE,SETT                        : BOOLEAN;
      DAY,MONTH,YEAR, I,IOBJ,NZ              : INTEGER;
      LAMBDA,ZONE,PHI,SPHI,CPHI              : REAL,
      TSTART,DATE,HOUR,HH,UTRISE,UTSET       : REAL;
      Y_MINUS,Y_0,Y_PLUS,ZERO1,ZERO2,XE,YE: REAL;
      SINH0                                  : ARRAY[1..3] OF REAL;
(*-----------------------------------------------------------------*)
```

```
(* Сюда надо ввести подпрограммы в такой последовательности:          *)
(*    SN, CS, QUAD                                                     *)
(*    MJD, CALDAT, LMST                                                *)
(*    MINISUN, MINIMOON                                                *)
(*-------------------------------------------------------------------- *)
(*-------------------------------------------------------------------- *)
(* SIN_ALT:   sin(высоты над горизонтом)                               *)
(*            IOBJ:   1 = Луна, 2 = Солнце                             *)
(*-------------------------------------------------------------------- *)
FUNCTION SIN_ALT(IOBJ:INTEGER;
MJD0,HOUR,LAMBDA,CPHI,SPHI:REAL): REAL;
  VAR MJD,T,RA,DEC,TAU: REAL;
  BEGIN
    MJD:= MJD0 + HOUR/24.0;
    T  := (MJD-51544.5)/36525 0;
    IF  (IOBJ=1)
      THEN MINI_MOON(T,RA,DEC)
      ELSE  MINI_SUN  (T,RA,DEC);
    TAU    :=  15.0 * (LMST(MJD,LAMBDA) - RA);
    SIN_ALT:= SPHI*SN(DEC) + CPHI*CS(DEC)*CS(TAU);
  END;
(*-------------------------------------------------------------------- *)
(* WHM: выдает время в часах и минутах                                 *)
(*-------------------------------------------------------------------- *)
PROCEDURE WHM(UT:REAL);
  VAR H,M:INTEGER;
  BEGIN
 UT:= TRUNC(UT*60.0+0.5)/60.0; (* округление до 1 мин                  *)
 H := TRUNC(UT);   M:=TRUNC(60.0*(UT-H)+0.5);   WRITE(H:5,':',M:2,'   ');
  END;
(*-------------------------------------------------------------------- *)
(* GETINP: читает и выводит исходные данные                            *)
(*-------------------------------------------------------------------- *)
PROCEDURE GETINP(VAR DATE,LAMBDA,PHI,ZONE: REAL);
  VAR D,M,Y: INTEGER;
  BEGIN
    WRITELN;
    WRITELN('  SUNSET. моменты восхода и захода Солнца и Луны   ');
    WRITELN('                 Версия 28.12.88                   ');
    WRITELN('     (c) 1988 Thomas Pfleger, Oliver Montenbruck   ');
    WRITELN;
    WRITELN;
    WRITE   (' Начальная дата (DD MM YYYY)              ...     ');
    READLN(D,M,Y);
    WRITELN;
    WRITE   ('Место наблюдения: долгота (восточная—отриц.)...   ');
    READLN(LAMBDA);
```

```
    WRITE   ('                       широта                   ....   ');
    READLN(PHI);
    WRITE   ('             поясное время — UT (h)             ...    ');
    READLN(ZONE);
    WRITELN;
    WRITELN;
    WRITELN('   Дата         Луна          Солнце          Сумерки ');
    WRITELN;
    WRITELN('          восх./зах      восх./зах.      конец/начало '),
    WRITELN;
    ZONE.=  ZONE /24.0;
    DATE:=  MJD(D,M,Y,0)-ZONE;
  END,
(*------------------------------------------------------------------      *)
BEGIN (* SUNSET *)
  SINHO[1]  :=  SN  ( +8 0/60 0),     (* восход Луны      при  h= +8'     *)
  SINHO[2]  :=  SN  (-50.0/60 0),     (* восход Солнца    при  h=-50'     *)
  SINHO[3]  :=  SN  (    -12 0   ),   (* нав  сумерки     при  h=-12 град *)
  GETINP (TSTART, LAMBDA, PHI, ZONE);
  SPHI := SN(PHI);     CPHI  :=  CS(PHI);
  FOR I:= 0 TO 9 DO (* цикл по дням в заданном промежутке                 *)
    BEGIN
     DATE:= TSTART  +  I;
     CALDAT(DATE+ZONE, DAY, MONTH, YEAR, HH);
     WRITE(DAY:3,'.',MONTH 2,'.',YEAR·4,' ');
                        (* выводится фактическая дата       *)
    FOR IOBJ  :=  1 TO 3 DO
     BEGIN
       HOUR   :=  1.0;
       Y_MINUS:= SIN_ALT(IOBJ, DATE, HOUR-
                    1.0, LAMBDA, CPHI, SPHI)-SINHO[IOBJ];
       ABOVE := (Y_MINUS>0.0);  RISE := FALSE;  SETT := FALSE;
(* цикл по исследуемому интервалу от [0h-2h] до [22h-24h]                 *)
 REPEAT
    Y_0    :=  SIN_ALT(IOBJ, DATE, HOUR, LAMBDA, CPHI, SPHI)-SINHO[IOBJ];
    Y_PLUS:=  SIN_ALT(IOBJ, DATE, HOUR+
                1.0, LAMBDA, CPHI, SPHI)-SINHO[IOBJ];
(* поиск параболы, проходящей через три значения                          *)
(*                        Y_MINUS, Y_0, Y_PLUS                            *)
        QUAD ( Y_MINUS, Y_0, Y_PLUS,  XE, YE,  ZERO1, ZERO2,  NZ );
          CASE (NZ) OF
             0:  ;
             1:  IF  (Y_MINUS<0.0)
                 THEN BEGIN UTRISE:=HOUR+ZERO1;   RISE:=  TRUE; END
                 ELSE BEGIN UTSET:=  HOUR+ZERO1;   SETT:=  TRUE; END;
             2:  BEGIN
                     IF   (YE<0.0)
```

```
          THEN BEGIN UTRISE:=HOUR+ZERO2; UTSET:=HOUR+ZERO1; END
          ELSE BEGIN UTRISE:=HOUR+ZERO1; UTSET:=HOUR+ZERO2; END;
                 RISE:= TRUE, SETT:= TRUE;
               END;
            END;
    Y_MINUS:=Y_PLUS;             (* таблица для              *)
          HOUR:=HOUR + 2.0;      (* следующего интервала     *)
       UNTIL ( (HOUR=25.0) OR (RISE AND SETT) );
    IF (RISE OR SETT) (* вывод *)
      THEN
        BEGIN
          IF RISE THEN WHM(UTRISE) ELSE WRITE('------':9    );
          IF SETT THEN WHM(UTSET)  ELSE WRITE('------':9    );
            END
          ELSE
          BEGIN
            IF ABOVE
              THEN CASE IOBJ OF
                   1,2: WRITE ('   не заходит '           );
                   3:   WRITE ('   постоянно светло    ');
                  END
              ELSE CASE IOBJ OF
                     1,2: WRITE ('   не восходит      '    );
                     3.   WRITE ('   постоянно темно  '   ');
                    END,
          END;
       END;
    WRITELN;
  END; (* конец цикла по дням в исследуемом промежутке времени   *)
  WRITELN; WRITE ('  время поясное ( = UT ');
  IF ZONE>=0 THEN WRITE('+'); WRITELN(ZONE*24.0:5:1,'h )');
END.
(*-----------------------------------------------------------*)
```

Мы хотим на примерах показать работу программы SUNSET и при этом обратить внимание на те особенности, которые могут встретиться при вычислении моментов восхода и захода.

SUNSET вычисляет моменты восхода и захода в интервале 10 сут, начинающемся с заданной даты. Географическая долгота места наблюдения к востоку от Гринвича, например в центральной Европе, вводится с отрицательным знаком. Наконец, задается разность поясного и всемирного времени в часах: 1 означает, например, среднеевропейское время (CET), 2 — среднеевропейское летнее время (CEST), а если нужно выдать данные в единицах всемирного времени, задают значение 0. Поясное время Северной Америки имеет отрицательные значения, например −5 для восточного стандартного времени (EST) или −8 для тихоокеанского стандартного времени (PST).

Вычислим моменты восхода и захода Солнца и Луны для Мюнхена (λ = 11,6° в.д., φ = 48,1° с.ш.) 23 марта 1989 г. Результат мы хотим получить в единицах среднеевропейского времени (отличается от всемирного времени на 1^h). В первую очередь нужно задать момент начала 10-дневного промежутка времени в форме: день, месяц, год. После этого программа запрашивает географические координаты и разность поясных времен. Все вводимые данные этого диалога представлены курсивным шрифтом Используя эти данные, SUNSET определяет моменты восхода и захода Луны, а также начало и конец сумерек. Поскольку в программе началу сумерек соответствует высота Солнца, равная −12°, все данные в этих примерах относятся к *навигационным* сумеркам

```
SUNSET: Моменты восхода и захода Солнца и Луны
Версия 28.12 88
(c) 1988 Thomas Pfleger, Oliver Montenbruck

Начальная дата (DD MM YYYY)                          ...  23 3 1989
место наблюдения  долгота (восточная—отриц )         ...  -11.6
                  широта                              ...  48.1
                  поясное время-UT (h)                ...  1.0
```

Дата	Луна восх.	Луна зах.	Солнце восх.	Солнце зах.	Сумерки конец	Сумерки начало
23. 3 1989	19:57	6:13	6:11	18:30	5: 3	19:39
24. 3.1989	21· 5	6:28	6: 9	18:32	5: 1	19:40
25. 3.1989	22:15	6:45	6: 7	18·33	4:59	19:42
26. 3.1989	23:26	7: 6	6: 5	18:35	4:56	19:44
27 3.1989	-----	7·33	6: 2	18:36	4:54	19:45
28 3.1989	0:34	8: 9	6: 0	18:38	4:52	19:47
29. 3.1989	1:38	8:58	5:58	18:39	4:50	19:48
30 3.1989	2:31	10 0	5:56	18:41	4:48	19:50
31 3.1989	3:14	11:14	5:54	18:42	4:45	19:52
1 4.1989	3:47	12:35	5:52	18:43	4·43	19:53

время поясное (= UT + 1h)

В приведенной таблице обращает на себя внимание столбец восходов Луны, здесь 27.3 момент восхода отсутствует В эту ночь Луна восходит после полуночи, и потому восход приходится на 28.3. Обычно в каждом месяце имеется такой день, в который Луна не восходит или не заходит.

Для ознакомления с другим интересным явлением вычислим моменты восхода и захода для некоторого места в Европе на широте 65°. Начальной датой 10-дневного периода примем 15 июля 1989 г., а поясным временем будем считать летнее среднеевропейское время. По программе SUNSET получим

```
SUNSET: Моменты восхода и захода Солнца и Луны
Версия 28 12.88.
(c) 1988 Thomas Pfleger, Oliver Montenbruck
```

Начальная дата (DD MM YYYY)	...	*15.6.1989*
место наблюдения: долгота (восточная – отриц.)	...	*-10.0*
широта	...	*65.0*
поясное время – UT (h)	...	*2.0*

Дата	Луна восх./зах.		Солнце восх./зах.		Сумерки конец/начало
15. 6.1989	19:58	1: 0	2:24	0:16	постоянно светло
16. 6.1989	22:26	23:53	2:23	0:18	постоянно светло
17. 6.1989	не видно		2:22	0:19	постоянно светло
18. 6.1989	не видно		2:21	0.18	постоянно светло
19. 6.1989	не видно		2:20	0:21	постоянно светло
20. 6.1989	не видно		2:20	0:22	постоянно светло
21. 6.1989	2:39	3:24	2:20	0:23	постоянно светло
22. 6.1989	1:35	6:21	2·20	0:23	постоянно светло
23. 6.1989	1:15	8:29	2:21	0:23	постоянно светло
24. 6.1989	1: 1	10.25	2·22	0:22	постоянно светло

время поясное (= UT + 2h)

Здесь бросается в глаза, что Луна после ее захода 16 июля в 23^h53^m остается под горизонтом до 2^h39^m 21 июня, и поэтому она не видна. Точно так же на высоких широтах Луна несколько дней может оставаться над горизонтом.

Разумеется, подобные условия видимости характерны и для Солнца. В этом случае говорят о полярном дне или полярной ночи. При помощи программы SUNSET можно для определенного места вычислить, например, продолжительность полярной ночи. При этом временной интервал в вычислениях меняется до тех пор, пока для Солнца впервые не будет достигнуто условие «не видно». В эти сутки и начинается полярная ночь. Окончание полярной ночи определяют таким же способом.

Результат «постоянно светло» в столбце сумерек означает, что уже вообще не темнеет. Такие условия возможны, начиная с широты 50°, если речь идет об астрономических сумерках.

4. ОРБИТЫ КОМЕТ

В то время как орбиты больших планет вычисляют и публикуют на годы вперед, предсказания орбит комет в большинстве случаев ограничиваются краткосрочными эфемеридами и указанием элементов орбиты. Пять-шесть элементов описывают пространственное и временное поведение орбиты кометы намного компактнее, чем таблица положений. Орбиты вновь открытых комет, как правило, публикуются в циркуляре Международного астрономического союза (МАС). Наряду с этим существует ряд каталогов, в которых приведены данные для периодических комет и астероидов.

Имея элементы орбиты, с помощью соответствующей программы можно определить положение кометы относительно Земли и Солнца. Программа COMET, представленная в этой главе, позволяет вычислять не только эллиптические орбиты планет и астероидов, но также параболические, почти параболические и гиперболические орбиты комет

В следующем разделе описывается движение кометы относительно Солнца и Земли без учета возмущений, обусловленных планетами, требующего больших дополнительных усилий Преобразования координат необходимые для определения геоцентрических экваториальных коорди нат, были описаны в гл.2 Перечислим подпрограммы, используемые в программе COMET:

SN, CS, ATN, ATN2, CART, POLAR, GMS, CURB,
MJD, CALDAT
ECLEQU, PMATECL, PRECART,
SUN200.

4.1. Форма и положение орбиты

Движение комет вокруг Солнца происходит по тем же законам, что и движение планет.

1. Орбита имеет форму конического сечения — эллипса, параболы или гиперболы, — в одном из фокусов которого расположено Солнце. Это, в частности, означает, что орбита лежит в неподвижной плоскости.
2. За равные промежутки времени линия, соединяющая Солнце и комету, заметает равные площади (*закон площадей*)

Эти законы были сформулированы Иоганном Кеплером и позднее бы сведены Ньютоном к общему закону тяготения. Правда, действуют зак ны Кеплера только в том случае, когда влияние планет на кометы п небрежимо мало по сравнению с силой притяжения Солнца.

Сила тяготения Солнца всегда действует вдоль линии, соединяющ Солнце и комету (центральная сила). Этим и объясняется неподви ность плоскости, в которой лежит орбита Двигаясь со скоростью $\dot{\mathbf{r}}$, комета за небольшой промежуток времени Δt пройдет отрезок $\dot{\mathbf{r}} \cdot \Delta t$ от положения $\mathbf{r}$ до положения $\mathbf{r} + \dot{\mathbf{r}} \cdot \Delta t$ (рис.4.1). Эти два положения вместе с положением Солнца определяют ориентацию плоскости орбиты в пространстве. Так как тяготение Солнца действует на комету только в этой плоскости, движение кометы ограничивается ею. Закон площадей также является прямым следствием действия центральной силы. Доказательство этого положения требует математических выкладок, и мы не будем его приводить.

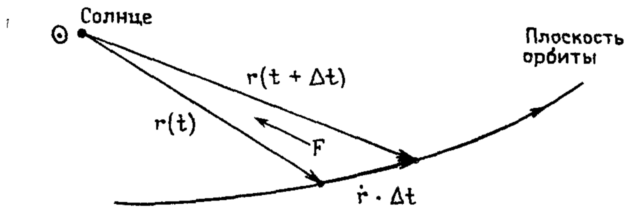
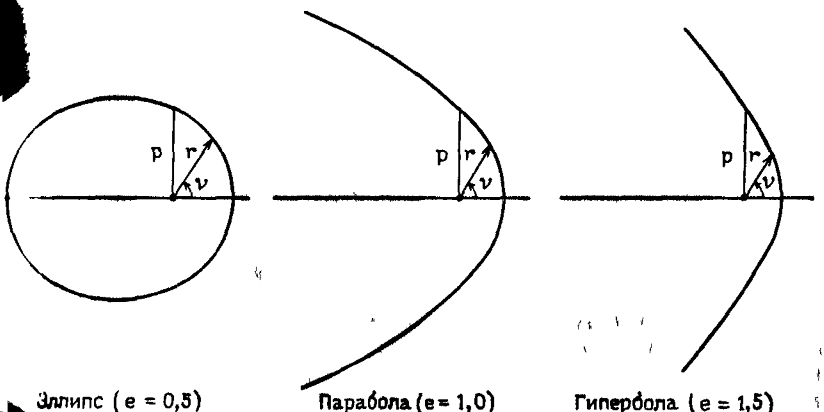


Рис.4.1. Сила, действующая на комету, не меняет плоскость орбиты.

Точная форма орбиты определяется квадратическим ослаблением силы тяготения с увеличением расстояния ($F \propto 1/r^2$). На рис.4.2 представлены три основные формы орбиты—эллипс, парабола и гипербола Движение по эллиптическим орбитам всегда периодическое, но кометы, имеющие параболические и гиперболические орбиты, приближаются к



.2. Конические сечения с одинаковым параметром орбиты p и нтриситетом e = 0,5, e = 1,0 и e = 1,5.

Солнцу лишь один раз, причем на короткое время. Ближайшая к Солнцу точка орбиты называется *перигелием*. Линия, соединяющая Солнце и перигелий (*линия апсид*), делит орбиту на две симметричные половины.

Положение тела на орбите определяется расстоянием от Солнца r и *истинной аномалией* ν[1], представляющей собой угол между направлением Солнце—комета и направлением Солнце—перигелий. Взаимосвязь между r и ν дается уравнением конического сечения:

$$r = \frac{p}{1 + e\cos(\nu)}, \tag{4.1}$$

где p—расстояние от Солнца при $\nu = 90°$, называемое *параметром орбиты*, определяет размер орбиты. *Эксцентриситет* орбиты e показывает, насколько сильно отличается форма орбиты от окружности ($e = 0$). С увеличением эксцентриситета эллипс вытягивается и в конце концов превращается в параболу ($e = 1$), а затем в гиперболу ($e > 1$).

Вместо параметра орбиты часто употребляется большая полуось a, которая определена только для $e \neq 1$:

$$a = \frac{p}{1 - e^2}, \qquad p = a(1 - e^2). \tag{4.2}$$

В случае эллиптической орбиты a имеет значение, равное среднему между наибольшим и наименьшим расстояниями от Солнца:

$$r_{\min} + r_{\max} = \frac{p}{1+e} + \frac{p}{1-e} = \frac{2p}{1-e^2} = 2a.$$

Нужно отметить, что при гиперболической орбите, когда, согласно определению, $e > 1$, большая полуось имеет отрицательное значение. В случае орбит, близких к параболическим ($e \approx 1$), вместо a используют расстояние до перигелия $q = r_{\min}$:

$$q = \frac{p}{1+e}. \tag{4.3}$$

Для параболической орбиты $q = p/2$.

4.2. Положение на орбите

Хотя уравнение конического сечения дает все возможные положения кометы на орбите во время ее движения, оно ничего не говорит о том, каким моментам времени соответствуют эти положения. Для решения этой задачи используют закон площадей. Основные этапы вычислений мы попытаемся объяснить на примере эллиптической орбиты.

[1] Под аномалией в небесной механике понимают угол. Помимо истинной аномалии применяются средняя и эксцентрическая аномалии.

Эллипс с эксцентриситетом e можно получить, сжав круг в одном направлении в $\sqrt{1-e^2}$ раз. Площадь эллипса S с большой полуосью a при этом во столько же раз меньше площади круга с радиусом a:

$$S = (\pi a^2)\sqrt{1-e^2}\ .$$

Линия, соединяющая Солнце и комету (радиус-вектор), заметает полный эллипс за время обращения T, а сектор площадью

$$S(t) = (\pi a^2)\sqrt{1-e^2}\left[\frac{t-t_0}{T}\right] \tag{4.4}$$

—к моменту t после прохождения перигелия в момент t_0. Теперь, если удастся представить площадь этого сектора S с помощью координат (r, ν) точки, в которой находится комета на орбите, мы найдем искомую зависимость между положением на орбите и временем t. В случае круговой орбиты это очень просто, так как S здесь прямо пропорционально углу ν и квадрату полуоси. В случае эллипса нужно ввести вспомогательную величину, называемую *эксцентрической аномалией*. Ее геометрический смысл становится ясен при рассмотрении рис.4.3. Если ось x направить так, как на рисунке, то точка P на эллипсе будет иметь x-координату $x = a \cdot \cos E - a \cdot e$. Здесь член $a \cdot e$ определяет расстояние от центра эллипса до фокуса, в котором расположено Солнце. У точки P y-координата в $\sqrt{1-e^2}$ раз меньше, чем y-координата точки P' на окружности, и равняется $y = \sqrt{1-e^2} \cdot a \cdot \sin E$.

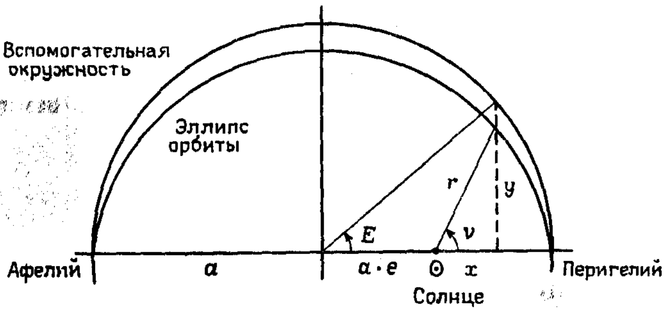


Рис.4.3. Определение эксцентрической аномалии E.

Таким образом, получим:

$$\begin{aligned} x &= r\cos\nu = a(\cos E - e),\\ y &= r\sin\nu = a\sqrt{1-e^2}\sin E. \end{aligned} \tag{4.5}$$

Для вычисления S рассмотрим вспомогательные фигуры, изображенные на рис.4.4.

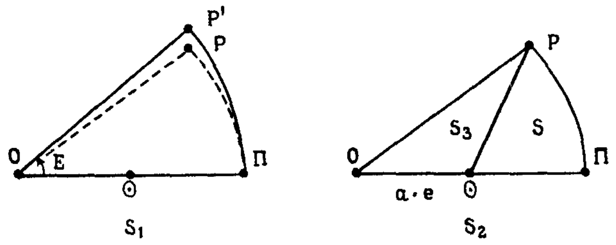


Рис.4.4. К определению площади сектора S

Сектор круга ($P'O\Pi$):	$S_1 = \pi\ a^2\ E/360°$
Сектор эллипса ($PO\Pi$):	$S_2 = \sqrt{1 - e^2} \cdot S_1$
Треугольник ($O \circ P$):	$S_3 = \frac{1}{2}\ y\ (a\ e) = \frac{1}{2}\ a^2\ e \sqrt{1 - e^2}\ \sin E$
Сектор ($P \circ \Pi$):	$S = S_2 - S_3$.

Используя эксцентрическую аномалию, можно представить площадь S в виде

$$S(t) = \frac{1}{2}\ a^2\sqrt{1 - e^2}\ \left[\ E\ \frac{\pi}{180°} - e\ \sin\ E\ \right]. \tag{4 6}$$

Сравнивая эту формулу с вышеприведенной для $S(t)$, можно получить зависимость между эксцентрической аномалией и временем, называемую *уравнением Кеплера:*

$$E(t) - \frac{180°}{\pi} e\ \sin\ E(t) = M(t), \tag{4.7}$$

где

$$M(t) = 360°\ \frac{(t - t_0)}{T}\ . \tag{4.8}$$

Чтобы вычислить положение кометы на орбите в любой момент, нужно это уравнение решить относительно E и полученное значение подставить в уравнение (4.5) для определения x и y, как подробно описано в разд.4 3, посвященном численному решению уравнения Кеплера.

До сих пор время обращения по орбите рассматривалось как независимая величина. На самом деле, согласно третьему закону Кеплера, оно связано с величиной большой полуоси орбиты:

$$\frac{a^3}{T^2} = \frac{GM_\odot}{4\pi^2}, \quad \text{где} \quad GM_\odot = 1{,}32712 \cdot 10^{20}\ \text{м}^3 \cdot \text{с}^{-2}. \tag{4.9}$$

Константа в правой части уравнения содержит произведение гравитационной постоянной и массы Солнца. Единицы — метр и секунда, через которые выражены эти величины, в практике таких расчетов не используются. Обычными единицами измерения являются сутки (1 сут = 86400 с) и астрономическая единица (1 а е $\simeq$ 149,59787 млн км). Вначале эта единица определялась величиной большой полуоси орбиты Земли вокруг Солнца. Позднее Международный астрономический союз принял решение определить астрономическую единицу таким образом, чтобы произведение гравитационной постоянной на массу Солнца имело фиксированное значение

$$GM_\odot = k^2\ \text{а е}^3\ \text{сут}^{-2}, \quad \text{где}$$

$$k = 0{,}01720209895,$$

$$k^2 \simeq 2{,}959122083 \cdot 10^{-4}.$$

Величину k называют гравитационной постоянной Гаусса. Хотя на первый взгляд приведенное определение кажется необычным, благодаря ему величина астрономической единицы не зависит от возможных вековых изменений диаметра орбиты Земли

Наряду с положением кометы на орбите необходимо знать и ее скорость Приведем формулы, определяющие эту величину

$$\dot{x} = -\sqrt{\frac{GM_\odot}{a}} \cdot \frac{\sin E}{1 - e\cos E},$$
$$\dot{y} = +\sqrt{\frac{GM_\odot(1-e^2)}{a}} \cdot \frac{\cos E}{1 - e\cos E}. \tag{4.10}$$

Такие же формулы существуют и для гиперболической орбиты, однако вместо угловых функций в них фигурируют гиперболические функции sh $x = (e^x - e^{-x})/2$ и ch $x = (e^x + e^{-x})/2$. Получим

$$x = r\cos\nu = |a|\,(e - \operatorname{ch} H),$$
$$y = r\sin\nu = |a|\sqrt{e^2-1}\ \operatorname{sh} H \tag{4.11}$$

$$\dot{x} = -\sqrt{\frac{GM_\odot}{|a|}}\ \frac{\operatorname{sh} H}{e\operatorname{ch} H - 1},$$
$$\dot{y} = +\sqrt{\frac{GM_\odot(e^2-1)}{|a|}}\ \frac{\operatorname{ch} H}{e\operatorname{ch} H - 1}, \tag{4.12}$$

Величина H, соответствующая эксцентрической аномалии, удовлетворяет модифицированной форме уравнения Кеплера

$$e \operatorname{sh} H - H = M_h = \sqrt{\frac{GM_{\odot}}{|a|^3}}\,(t - t_0). \qquad (4.13)$$

При параболической орбите ($e = 1$) x и y выражаются через $\operatorname{tg}(\nu/2)$:

$$x = r\cos\nu = q\left[1 - \operatorname{tg}^2[\nu/2]\right],$$

$$y = r\sin\nu = 2q\operatorname{tg}(\nu/2). \qquad (4.14)$$

Здесь q—уже знакомое нам перигелийное расстояние. Истинная аномалия ν связана с временем через *уравнение Баркера*:

$$\operatorname{tg}(\nu/2) + 1/3\operatorname{tg}^3(\nu/2) = \sqrt{\frac{GM_{\odot}}{2q^3}}\,(t - t_0). \qquad (4.15)$$

В это уравнение $\operatorname{tg}(\nu/2)$ входит в третьей степени. В отличие от уравнения Кеплера здесь мы прямо можем определить ν через заданное время t. Если положим

$$A = \frac{3}{2}\sqrt{\frac{GM_{\odot}}{2q^3}}\,(t - t_0) \quad \text{и} \quad B = \sqrt[3]{A + \sqrt{A^2 + 1}},$$

то получим

$$\operatorname{tg}(\nu/2) = B - 1/B, \qquad \nu = 2\operatorname{arctg}(B - 1/B).$$

4.3. Численное решение уравнения Кеплера

Аналитическое решение уравнения Кеплера относительно эксцентрической аномалии как функции средней аномалии или времени невозможно, поэтому для получения решения используют численные методы.

В первую очередь для заданного времени t вычисляют среднюю аномалию M. При небольших эксцентриситетах она мало отличается от эксцентрической аномалии E. Поэтому M можно использовать как E_0—первое приближение для последующей итерации. Немного преобразовав уравнение Кеплера, мы увидим, что для его решения нужно определить нуль функции

$$f(E) = E - \frac{180^\circ}{\pi}\, e\sin E - M(t).$$

Для определения нуля функции $f(x)$ чаще всего применяют *метод Ньютона* (рис.4.5). В точке x, близкой к нулю, кривую функции заменяют касательной с наклоном $f'(x)$ и находят точку ее пересечения с осью абсцисс:

$$\hat{x} = x - \frac{f(x)}{f'(x)} .$$

Таким образом получают улучшенное значение $\hat{x}$ для нуля функции f. Итерация для уравнения Кеплера будет иметь вид

$$\hat{E} = E - \frac{E - (180°/\pi)\ e \sin E - M(t)}{1 - e \cos E} .$$

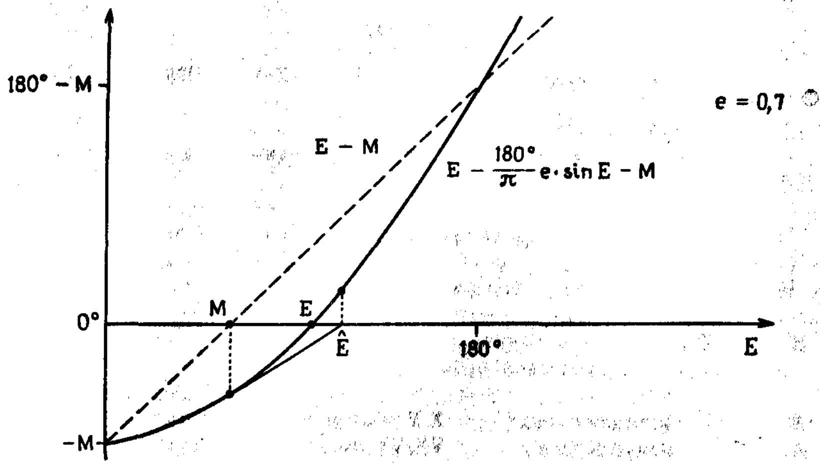


Рис.4.5. Решение уравнения Кеплера методом Ньютона.

Здесь E выражено в градусах. Если в вычислениях используется радианная мера, то множитель $180°/\pi$ заменяют единицей. При орбите, близкой к круговой, обычно за три итерации получают приблизительно десять разрядов в точном значении E. Но нужно быть очень осмотрительным, когда метод Ньютона используется при больших эксцентриситетах. Мы рекомендуем читателю самому убедиться в этом, рассмотрев ход итерации при $e = 0{,}99$ и $E_0 = M = 5°$. Для надежности лучше использовать другое начальное значение, а именно $E_0 = \pi = 180°$. В процедуре ECCANOM это начальное значение используется для эксцентриситетов, превышающих $e = 0{,}8$.

```
(*-----------------------------------------------------------------*)
(*ECCANOM:Вычисление эксцентрической аномалии E=ECCANOM(MAN,ECC)*)
(*          по средней аномалии MAN и эксцентриситету ECC.        *)
(*          (Решение уравнения Кеплера методом Ньютона)           *)
(*          (E, MAN в градусах)                                   *)
(*-----------------------------------------------------------------*)
FUNCTION ECCANOM(MAN,ECC:REAL):REAL;
  CONST PI=3 141592654;  TWOPI=6 283185308;  RAD=0 0174532925199433,
        EPS = 1E-11;  MAXIT = 15;
  VAR M,E,F: REAL;
      I    : INTEGER;
  BEGIN
    M:= MAN/360.0;  M:=TWOPI*(M-TRUNC(M)); IF M<0 THEN M:=M+TWOPI,
    IF  (ECC<0.8) THEN E:=M ELSE E:=PI;
    F:= E - ECC*SIN(E) - M;  I:=0;
    WHILE ( (ABS(F)>EPS) AND (I<MAXIT) ) DO
      BEGIN
        E:= E - F / (1 0-ECC*COS(E));  F: = E-ECC*SIN(E)-M;  I:=I+1;
      END;
    ECCANOM.=E/RAD;
    IF (I=MAXIT) THEN  WRITELN('проблема сходимости в ECCANOM'),
  END;
(*-----------------------------------------------------------------*)
```

Теперь можно определять эксцентрическую аномалию одним обращением к этой процедуре. Остальные действия, необходимые для вычисления положения тела на орбите, содержатся в процедуре ELLIP.

```
(*-----------------------------------------------------------------*)
(* ELLIP:  Вычисление векторов положения и скорости               *)
(*         для эллиптической орбиты                               *)
(*                                                                *)
(* M средняя аномалия (град.)     X,Y вектор положения (а.е.)     *)
(* A большая полуось (а.е.)       VX,VY скорость (а.е./сут)       *)
(* ECC эксцентриситет                                             *)
(*-----------------------------------------------------------------*)
PROCEDURE ELLIP(M,A,ECC:REAL;VAR X,Y,VX,VY:REAL);
  CONST KGAUSS = 0.01720209895;
  VAR K,E,C,S,FAC,RHO. REAL;
  BEGIN
    K  : = KGAUSS / SQRT(A);
    E  : = ECCANOM(M,ECC); C.=CS(E);  S:=SN(E);
    FAC: = SQRT((1.0-ECC)*(1+ECC));     RHO:=1.0-ECC*C;
    X  : = A*(C-ECC); Y :=A*FAC*S;      VX:=-K*S/RHO; VY:=K*FAC*C/RHO;
  END;
(*-----------------------------------------------------------------*)
```

Таким же образом решается уравнение Кеплера для гиперболической орбиты. Здесь итерация выглядит так:

$$\hat{H} = H - \frac{e \operatorname{sh} H - H - M_h(t)}{e \operatorname{ch} H - 1} .$$

Начальными значениями здесь будут

$$H_0 = \begin{cases} +\ln(+1{,}8 + 2M_h/e) & \text{при} \quad M_h \geq 0, \\ -\ln(-1{,}8 - 2M_h/e) & \text{при} \quad M_h < 0 \end{cases}$$

```
(*-----------------------------------------------------------*)
(* HYPANOM:Вычисление эксцентрической аномалии H=HYPANOM(MH,ECC)*)
(*         для средней аномалии MH и эксцентриситета ECC        *)
(*         гиперболической орбиты                               *)
(*-----------------------------------------------------------*)
FUNCTION HYPANOM(MH,ECC.REAL):REAL;
  CONST EPS=1E-10, MAXIT=15,
  VAR   H,F,EXH,SINHH,COSHH· REAL,
        I                  : INTEGER,
  BEGIN
   H    : =LN(2 0*ABS(MH)/ECC+1.8); IF (MH<0.0) THEN H:=-H;
   EXH  : =EXP(H),
   SINHH: =0.5*(EXH-1.0/EXH);
   COSHH: =0.5*(EXH+1.0/EXH);
   F    : ECC*SINHH-H-MH; I =0;
   WHILE ( (ABS(F)>EPS*(1.0+ABS(H+MH))) AND (I<MAXIT) )  DO
     BEGIN
       H    := H - F / (ECC*COSHH-1.0);
       EXH  := EXP(H);
       SINHH:= 0.5*(EXH-1.0/EXH);
       COSHH:=0.5*(EXH+1.0/EXH);
       F    := ECC*SINHH-H-MH; I:=I+1;
     END;
   HYPANOM := H;
   IF (I=MAXIT) THEN  WRITELN(' проблема сходимости в HYPANOM ');
  END;
(*-----------------------------------------------------------*)
```

Прямоугольные координаты кометы на орбите можно вычислить с помощью процедуры HYPERB по формулам предыдущего раздела. В других случаях, как в ELLIP, приходится вводить вместо средней аномалии момент прохождения перигелия и момент, для которого ведется расчет. Выбор вводимых данных обусловлен тем, что ELLIP может быть использована и для вычисления орбит астероидов и планет. А у этих небесных тел в отличие от орбит комет момент прохождения перигелия очень редко используется как элемент орбиты. Вместо этого часто задают среднюю аномалию M_{Ep} и ее суточное изменение (*суточное движение*) n для определенной эпохи t_{Ep} . Средняя аномалия $M = M_{Ep} + n\,(t - t_{Ep})$ для любого момента t может быть прямо введена в подпрограмму ELLIP. Гиперболические орбиты встречаются только у комет и здесь в качестве элемента орбиты обычно используется время прохождения перигелия.

```
(*-----------------------------------------------------------------------*)
(* HYPERB: Вычисление векторов положения и скорости                      *)
(*         для гиперболической орбиты                                    *)
(*-----------------------------------------------------------------------*)
(* T0    момент прохождения перигелия    X,Y вектор положения            *)
(* T     текущее время                   VX,VY скорость                  *)
(* A     большая полуось (знак любой)                                    *)
(* ECC   эксцентриситет                                                  *)
(* (T0,T в юлианских столетиях с J2000)                                  *)
(*-----------------------------------------------------------------------*)
PROCEDURE HYPERB(T0,T,A,ECC:REAL;VAR X,Y,VX,VY:REAL);
  CONST KGAUSS = 0.01720209895;
  VAR K,MH,H,EXH,COSHH,SINHH,RHO,FAC: REAL;
  BEGIN
    A     : = ABS(A);
    K     : = KGAUSS / SQRT(A);
    MH    : = K*36525.0*(T-T0)/A;
    H     : = HYPANOM(MH,ECC);
    EXH   : = EXP(H);
    COSHH: =0.5*(EXH+1.0/EXH);
    SINHH: =0.5*(EXH-1.0/EXH);
    FAC   : = SQRT((ECC+1.0)*(ECC-1.0));    RHO := ECC*COSHH-1.0;
    X     : = A*(ECC-COSHH);   Y := A*FAC*SINHH;
    VX    : =-K*SINHH/RHO;     VY:= K*FAC*COSHH/RHO;
  END;
(*-----------------------------------------------------------------------*)
```

4.4. Орбиты, близкие к параболическим

Как видно из предыдущего раздела, вычисление эллиптических и гиперболических орбит комет с эксцентриситетом около единицы и малой аномалией — не простая задача. Эту трудность можно преодолеть, с самого начала предположив, что орбита близка к параболической. Небесный механик Карл Штумпф предложил следующий метод.

Из уравнения Кеплера

$$E(t) - e \sin E(t) = \sqrt{\frac{GM_\odot}{a^3}}\ (t - t_0),$$

которое здесь дано в радианной мере ($180^\circ = \pi$), при

$$q = a/(1 - e) \quad \text{и} \quad c_3 = (E - \sin E)/E^3$$

следует

$$(1 - e)\ E + e\ c_3(E)\ E^3 = \sqrt{GM_\odot \left[\frac{1 - e}{q}\right]^3}\ (t - t_0).$$

Введя новую переменную

$$U = \sqrt{\frac{3ec_3(E)}{1 - e}} \cdot E,$$

получим модифицированную форму:

$$U + \frac{1}{3} U^3 = \sqrt{6ec_3(E)} \cdot \sqrt{\frac{GM_\odot}{2q^3}} \cdot (t - t_0).$$

Это уравнение имеет формальный вид уравнения Баркера и при известной правой части может быть решено относительно U. Далее, при $c_1(E) = \sin(E)/E$ и $c_2(E) = (1 - \cos(E))/E^2$ получим

$$x = r\cos v = \frac{q}{1 - e}(\cos E - e) = q \cdot \left[1 - \left[\frac{2c_2}{6ec_3}\right] U^2\right],$$

$$y = r\sin v = \frac{q}{1 - e}\sqrt{1 - e^2}\sin E = 2q \cdot \sqrt{\frac{1 + e}{2e}}\left[\frac{1}{6c_3}\right] c_1 U = \tag{4.16}$$

$$= q \cdot \left[1 + \left[\frac{2c_2}{6c_3}\right] U^2\right].$$

Позднее понадобятся формулы для скорости кометы, вывод которых довольно трудоемок, и поэтому мы его не приводим.

$$\dot{x} = -\sqrt{\frac{GM_\odot}{q(1 + e)}}\left[\frac{y}{r}\right],$$

$$\dot{y} = +\sqrt{\frac{GM_\odot}{q(1 + e)}}\left[\frac{x}{r} + e\right]. \tag{4.17}$$

Кроме того, U и истинная аномалия v связаны **зависимостью**

$$\operatorname{tg}\left[\frac{v}{2}\right] = \left[\sqrt{\frac{1 + e}{3ec_3}}\;\frac{c_2}{c_1}\right] \cdot U,$$

которую мы приводим здесь для полноты изложения.

Во всех этих уравнениях члены, заключенные в квадратные скобки, при приближении E к 0 и e к 1 стремятся к единице. В этом предельном случае формулы преобразуются в представленные выше уравнения параболических орбит.

Данные преобразования, естественно, не препятствуют использованию метода итерации при решении уравнения Кеплера Однако при этом итерация не сопряжена с трудностями численного решения. Мы начнем с предположения, что $E \simeq 0$ ($c_3(E) \simeq 1/6$), и определим соответствующее значение U решением уравнения Кеплера—Баркера

$$U = B - 1/B,$$

где

$$B = \sqrt[3]{A + \sqrt{A^2 + 1}} \quad \text{и} \quad A = \frac{3}{2}\sqrt{6ec_3(E)}\sqrt{\frac{GM_\odot}{2q^3}}\,(t - t_0).$$

Это дает улучшенные значения

$$\hat{E} = U\sqrt[3]{\frac{1 - e}{3ec_3(E)}} \quad \text{и} \quad c_3(\hat{E}) = \frac{\hat{E} - \sin\hat{E}}{\hat{E}^2},$$

с помощью которых можно получить более точное значение $\hat{u}$ Эти действия повторяют до тех пор, пока U не станет постоянной в пределах требуемой точности При этом важно правильно вычислить значения функций $c_1(E)$, $c_2(E)$ и $c_3(E)$ при малых значениях E Из-за взаимного сокращения отдельных членов невозможен прямой расчет через тригонометрические функции. Поэтому для c_1, c_2 и c_3 используют разложение Тейлора·

$$c_1(E) = \frac{\sin E}{E} = 1 - \frac{E^2}{3!} + \frac{E^4}{5!} - \quad \ldots = \sum_{n=1}^{\infty}\alpha_n,$$

$$c_2(E) = \frac{1 - \cos E}{E^2} = \frac{1}{2!} - \frac{E^2}{4!} + \frac{E^4}{6!} - \ldots = \sum_{n=1}^{\infty}\beta_n, \tag{4.18}$$

$$c_3(E) = \frac{E - \sin E}{E^3} = \frac{1}{3!} - \frac{E^2}{5!} + \frac{E^4}{7!} - \ldots = \sum_{n=1}^{\infty}\gamma_n.$$

Все три ряда содержат только четные степени E и потому сходятся очень быстро. Для слагаемых удобно использовать следующие рекурсивные соотношения·

$$\alpha_1 = 1, \qquad \beta_n = \alpha_n \cdot \frac{1}{2n}, \qquad \gamma_n = \beta_n \cdot \frac{1}{2n + 1},$$

$$\alpha_{n+1} = -E^2 \cdot \gamma_n \qquad (n = 1,\ldots).$$

Они используются в программе STUMPFF до тех пор, пока слагаемые не станут меньше желаемой точности EPS.

```
(*-----------------------------------------------------------------   *)
(*  STUMPFF. Вычисление функций Штумпфа C1 = sin(E)/E ,               *)
(*           C2 = (1-cos(E))/(E**2) и C3 = (E-sin(E))/(E**3)          *)
(*           при E2=E**2                                              *)
(*           (E: эксцентрическая аномалия в радианах)                 *)
(*-----------------------------------------------------------------   *)
PROCEDURE STUMPFF(E2:REAL;VAR C1,C2,C3:REAL);
  CONST EPS=1E-12;
  VAR N,ADD· REAL;
  BEGIN
   C1:=0.0;  C2:=0.0;  C3:=0.0,  ADD:=1.0;  N:=1.0;
   REPEAT
      C1:=C1+ADD;  ADD.=ADD/(2.0*N);
      C2.=C2+ADD,  ADD =ADD/(2.0*N+1.0);
      C3.=C3+ADD,  ADD.=-E2*ADD;  N:=N+1.0;
   UNTIL ABS(ADD)<EPS,
  END;
(*-----------------------------------------------------------------*)
```

Прочие уравнения для параболических и близких к параболическим орбит комет сведены в программу PARAB, которая вызывается так же, как и программа HYPERB.

```
(*-----------------------------------------------------------------   *)
(* PARAB: Вычисление векторов положения и скорости для                *)
(*        параболических и близких к параболе орбит, по Штумпфу       *)
(*-----------------------------------------------------------------   *)
(* T0     момент прохождения перигелия     X,Y вектор положения       *)
(* T      текущее время                    VX,VY  скорость            *)
(* Q      расстояние до перигелия                                     *)
(* ECC    эксцентриситет                                              *)
(* (T0,T  в юлианских столетиях с J2000)                              *)
(*-----------------------------------------------------------------   *)
PROCEDURE PARAB(T0,T,Q,ECC:REAL;VAR X,Y,VX,VY:REAL),
  CONST EPS  = 1E-9;
       KGAUSS= 0.01720209895;
       MAXIT = 15;
  VAR E2,E20,FAC,C1,C2,C3,K,TAU,A,U,U2: REAL;
     R: REAL;
     I: INTEGER;
  BEGIN
   E2 := 0.0;    FAC:=0.5*ECC;     I:=0;
   K  := KGAUSS / SQRT(Q*(1.0+ECC));
   TAU:= KGAUSS * 36525.0*(T-T0);
   REPEAT
      I  := I+1;
      E20:= E2;
      A  := 1.5*SQRT(FAC/(Q*Q*Q))*TAU;  A:=CUBR(SQRT(A*A+1.0)+A);
      U  := A-1.0/A;  U2:=U*U;  E2:=U2*(1.0-ECC)/FAC;
```

```
    STUMPFF(E2,C1,C2,C3);  FAC:=3.0*ECC*C3;
  UNTIL  (ABS(E2-E20)<EPS)OR(I>MAXIT);
  IF (I=MAXIT) THEN  WRITELN('проблема сходимости в PARAB);
  R:=Q*(1.0+U2*C2*ECC/FAC);
  X:=Q*(1.0-U2*C2/FAC);             VY:= K*(X/R+ECC);
  Y:=Q*SQRT((1.0+ECC)/FAC)*U*C1;   VX:=-K*Y/R;
END;
(*-----------------------------------------------------------*)
```

4.5. Векторы Гаусса

Представленные выше программы позволяют вычислить положение кометы в плоскости орбиты. Для определения эклиптических координат кометы прежде всего необходимо определить положение орбиты относительно эклиптики. Для этого нужны еще три элемента орбиты (рис.4.6)

- i—*наклон орбиты*—угол между плоскостью орбиты и эклиптикой Угол больше 90° означает, что комета движется в направлении, обратном обращению планет вокруг Солнца
- Ω—*долгота восходящего узла*—угол между точкой весеннего равноденствия и той точкой орбиты, в которой комета пересекает эклиптику с юга на север
- ω—*аргумент перигелия*—угол между направлением восходящего узла и направлением на ближайшую к Солнцу точку орбиты

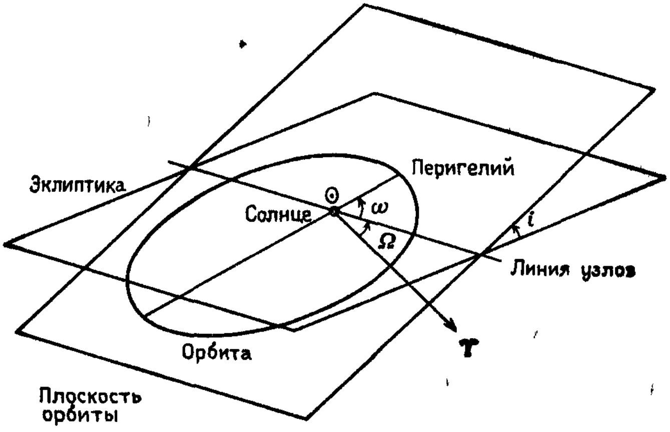


Рис.4.6. Орбитальные элементы кометы i, Ω и ω.

Вместо ω часто употребляют *долготу перигелия* ($\bar{\omega} = \Omega + \omega$), равную сумме долготы узла и аргумента перигелия.

Так как положение точки весеннего равноденствия и эклиптики медленно меняется из-за прецессии, элементы орбиты нужно задавать вместе с их равноденствием. «Равноденствие 1950» означает, например, что элементы вычислены по отношению к положению точки весеннего равноденствия и эклиптики 1950 г.

Исходной системой координат при выводе уравнений преобразования служит система координат (x'', y'', z''), ось x'' которой расположена вдоль направления на восходящий узел, а плоскость $x'' - y''$ совпадает с плоскостью орбиты (рис 4 7) Точка орбиты с истинной аномалией v и расстоянием от Солнца r имеет координаты

$$x'' = r \cos u,$$
$$y'' = r \sin u,$$
$$z'' = 0,$$

где *аргумент широты* u — сумма аргумента перигелия и истинной аномалии:

$$u = \omega + v. \qquad (4\ 19)$$

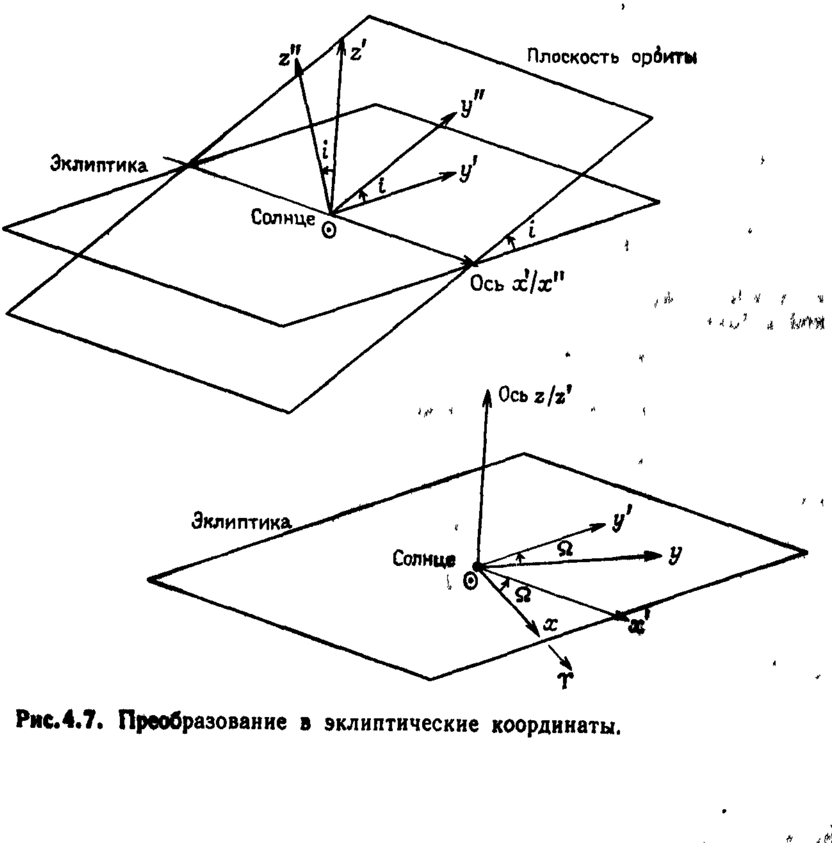


Рис.4.7. Преобразование в эклиптические координаты.

Теперь перейдем к системе (x', y', z'), ось x' которой также направлена вдоль линии узлов, а плоскость $x' - y'$ совпадает с эклиптикой. Переход между этими системами достигается поворотом на угол наклона орбиты i:

$$\begin{aligned} x' &= x'' &&= r\cos u, \\ y' &= y''\cos i - z''\sin i &&= r\sin u\cos i, \\ z' &= y''\sin i + z''\cos i &&= r\sin u\sin i. \end{aligned}$$

Полученные таким образом координаты относятся уже к эклиптике, но отсчитываются от восходящего узла. Поэтому следующим шагом будет изменение системы координат на такую, оси (x, y, z) которой правильно ориентированы относительно точки весеннего равноденствия и эклиптики:

$$\begin{aligned} x &= x'\cos\Omega - y'\sin\Omega, \\ y &= x'\sin\Omega + y'\cos\Omega, \\ z &= z'. \end{aligned}$$

Подставляя, получаем

$$\begin{aligned} x &= r\,(\cos u\cos\Omega - \sin u\cos i\sin\Omega), \\ y &= r\,(\cos u\sin\Omega + \sin u\cos i\cos\Omega), \\ z &= r\,(\sin u\sin i). \end{aligned} \tag{4.20}$$

Используя теорему сложения, заменяем

$$\cos u \text{ на } \cos\omega\cos\nu - \sin\omega\sin\nu \text{ и}$$

$$\sin u \text{ на } \sin\omega\cos\nu + \cos\omega\sin\nu$$

и получаем представление Гаусса, которое очень удобно в практической работе:

$$\begin{aligned} x &= r\cos\nu\,P_x + r\sin\nu\,Q_x, \\ y &= r\cos\nu\,P_y + r\sin\nu\,Q_y, \\ z &= r\cos\nu\,P_z + r\sin\nu\,Q_z \end{aligned} \tag{4.21}$$

при

$$\begin{aligned} P_x &= +\cos\omega\cos\Omega - \sin\omega\cos i\sin\Omega, \\ P_y &= +\cos\omega\sin\Omega + \sin\omega\cos i\cos\Omega, \\ P_z &= +\sin\omega\sin i \end{aligned} \tag{4.22}$$

и

$$Q_x = -\sin\omega\cos\Omega - \cos\omega\cos i\sin\Omega,$$

$$Q_y = -\sin\omega\sin\Omega + \cos\omega\cos i\cos\Omega, \tag{4.23}$$

$$Q_z = +\cos\omega\sin i.$$

Величины (P_x, P_y, P_z) и (Q_x, Q_y, Q_z) являются компонентами двух так называемых векторов Гаусса, **P** и **Q**. Оба вектора лежат в плоскости орбиты, перпендикулярны друг другу и имеют длину, равную единице, причем **P** направлен к перигелию. Векторы **P** и **Q** как раз образуют оси x и y системы координат, в которой выше рассматривалось движение кометы Для отличия от эклиптических координат $\mathbf{r} = (x, y, z)$ и $= (\dot{x}, \dot{y}, \dot{z})$ обозначим положение и скорость кометы на орбите соответственно через $(\tilde{x}, \tilde{y}) = (r\cos\nu, r\sin\nu)$ и $(\dot{\tilde{x}}, \dot{\tilde{y}})$ и получим

$$\mathbf{r} = \tilde{x}\,\mathbf{P} + \tilde{y}\,\mathbf{Q},$$

$$\mathbf{v} = \dot{\tilde{x}}\,\mathbf{P} + \dot{\tilde{y}}\,\mathbf{Q} \tag{4.24}$$

Применение этих соотношений особенно рекомендуется при вычислении целого ряда гелиоцентрических положений кометы Используемые при этом векторы Гаусса определяются один раз в начале вычислений, так как они зависят только от элементов орбиты, а не от времени.

```
(*-----------------------------------------------------------*)
(* GAUSVEC Вычисление векторов Гаусса (P,Q,R)               *)
(*         по эклиптическим элементам орбиты                 *)
(*         LAN = долгота восходящего узла                    *)
(*         INC = наклон орбиты                               *)
(*         AOP = аргумент перигелия                          *)
(*-----------------------------------------------------------*)
PROCEDURE GAUSVEC(LAN,INC,AOP:REAL;VAR PQR:REAL33);
  VAR  C1,S1,C2,S2,C3,S3:  REAL;
  BEGIN
    C1:=CS(AOP);    C2:=CS(INC);    C3.=CS(LAN);
    S1:=SN(AOP);    S2.=SN(INC);    S3 =SN(LAN);
    PQR[1,1]:=+C1*C3-S1*C2*S3;      PQR[1,2]:=-S1*C3-C1*C2*S3;
    PQR[1,3] =+S2*S3;               PQR[2,1]:=+C1*S3+S1*C2*C3;
    PQR[2,2]:=-S1*S3+C1*C2*C3;      PQR[2,3]:=-S2*C3;
    PQR[3,1].=+S1*S2;    PQR[3,2]:=+C1*S2;    PQR[3,3] =+C2;
  END;
(*-----------------------------------------------------------*)
(* ORBECL: Преобразование координат в системе плоскости орбиты *)
(*         в эклиптические координаты                         *)
(*-----------------------------------------------------------*)
PROCEDURE ORBECL(XX,YY:REAL;PQR REAL33;VAR X,Y,Z:REAL);
  BEGIN
```

```
    X:= PQR[1,1]*XX+PQR[1,2]*YY;
    Y:= PQR[2,1]*XX+PQR[2,2]*YY;
    Z:= PQR[3,1]*XX+PQR[3,2]*YY;
  END;
(*-----------------------------------------------------------------*)
```

Подпрограмма GAUSVEC позволяет по элементам орбиты вычислять векторы **P** и **Q** и перпендикулярный к ним вектор **R**. Все они имеют единичную длину. Хотя **R** здесь непосредственно не используется, его очень легко вычислить и он может пригодиться для решения других задач. PQR является матрицей 3 × 3, описанной как REAL 33. Соответствующее определение

```
TYPE REAL 33          : ARRAY [1..3,1..3];
```

должно быть добавлено в основную программу. С помощью программы ORBECL можно перейти от прямоугольных координат в плоскости орбиты к соответствующим трехмерным координатам, отнесенным к эклиптике

Рассмотренные до сих пор отдельные части программы вычисления движения комет могут быть объединены в одну подпрограмму, применимую для всех форм орбит, с помощью которой можно прямо получить эклиптические гелиоцентрические координаты.

```
(*-----------------------------------------------------------------*)
(* KEPLER: Вычисление векторов положения и скорости для             *)
(*         эллиптических, параболических и гиперболических орбит    *)
(*         (задача двух тел без учета возмущений)                   *)
(*                                                                  *)
(* T0      момент прохождения перигелия  X,Y,Z вектор положения     *)
(* T       текущее время                 VX,VY,VZ  скорость         *)
(* Q       расстояние перигелия                                     *)
(* ECC     эксцентриситет                                           *)
(* PQR     матрица векторов Гаусса                                  *)
(* (T0,T   в юлианских столетиях с J2000)                           *)
(*-----------------------------------------------------------------*)
PROCEDURE KEPLER(T0,T,Q,ECC:REAL;PQR:REAL33;
                 VAR X,Y,Z,VX,VY,VZ:REAL);
  CONST M0=5.0; EPS=0.1;
        KGAUSS = 0.01720209895; DEG = 57.29577951;
  VAR M,DELTA,TAU,INVAX,XX,YY,VVX,VVY: REAL;
  BEGIN
    DELTA:= ABS(1.0-ECC);
    INVAX:= DELTA / Q;
    TAU  := KGAUSS*36525.0*(T-T0);
    M    := DEG*TAU*SQRT(INVAX*INVAX*INVAX);
    IF ( (M<M0) AND (DELTA<EPS) )
      THEN PARAB(T0,T,Q,ECC,XX,YY,VVX,VVY)
        ELSE IF (ECC<1.0)
          THEN ELLIP (M,1.0/INVAX,ECC,XX,YY,VVX,VVY)
```

```
   ELSE HYPERB(T0,T,1.0/INVAX,ECC,XX,YY,VVX,VVY);
 ORBECL(XX,YY,PQR,X,Y,Z); ORBECL(VVX,VVY,PQR,VX,VY,VZ);
END;
(*------------------------------------------------------------*)
```

4.6. Время распространения света от кометы

Чтобы достигнуть наблюдателя, свету кометы в зависимости от расстояния требуется время от нескольких минут до нескольких часов В ечение этого времени сама комета перемещается в пространстве, потому наблюдаемое место расположения кометы не совпадает с ее ре-льным геометрическим местом. Скорость света c значительно больше арактерной скорости кометы, и учет времени распространения света т кометы в большинстве случаев дает поправку от нескольких секунд ю нескольких минут дуги Поэтому можно использовать приближения, прощающие определение времени распространения света от кометы. Чтобы вычислить видимое положение $\mathbf{r}(t)$ кометы в момент t, в первую очередь нужно для этого момента определить гелиоцентрические координаты кометы $\mathbf{r}_k(t)$ и геоцентрические координаты Солнца $\mathbf{r}_\odot(t)$ Величина

$$\Delta_0 = |\mathbf{r}_0| = |\mathbf{r}_\odot(t) + \mathbf{r}_k(t)| = \sqrt{(x_\odot + x_k^2) + (y_\odot + y_k^2) + (z_\odot + z_k^2)}$$

определяет для момента t геометрическое расстояние кометы от Земли. Оно мало отличается от расстояния, пройденного световым лучом

$$\Delta = |\mathbf{r}_\odot(t) + \mathbf{r}_k(t')| \qquad \text{при} \qquad t' = t - \frac{\Delta}{c},$$

оторое может быть вычислено только методом итераций, так как момент t' испускания света кометой изначально неизвестен. Итак, время аспространения света $\tau \simeq \Delta_0/c$, а гелиоцентрическое положение кометы в момент t' в хорошем приближении представляется как

$$\mathbf{r}_k(t') \simeq \mathbf{r}_k(t) - \mathbf{v}_k(t)\,\frac{\Delta_0}{c}\,.$$

Наблюдаемый геоцентрический вектор положения для «восстановленных» таким образом гелиоцентрических координат имеет вид

$$\mathbf{r} \simeq \mathbf{r}_\odot(t) + \left\{ \mathbf{r}_k(t) - \mathbf{v}_k(t)\,\frac{\Delta_0}{c} \right\}. \qquad (4.25)$$

Коэффициент $1/c$ в используемых нами единицах (а.е. и сут) имеет значение

$$1/c = 0{,}00578 \text{ сут/а.е.}$$

Исправленные таким образом координаты кометы называются *астрометрическими* координатами. Они удобны для непосредственного сравнения положения кометы и звездных координат на атласе неба Нужно отметить, что в качестве геоцентрического расстояния обычно приводят только значения геометрического расстояния Δ_0 (а не путь Δ, пройденный светом).

4.7. Программа COMET

Программа COMET позволяет вычислять эфемериды комет и астероидов методом решения задачи двух тел. При этом не учитываются возмущения, вызванные другими планетами Практически это упрощение не вносит заметных ошибок При сравнении результатов, вычисленных по этой программе, с другими данными следует учитывать, что они должны быть получены при одинаковых предположениях При использовании программы можно не выяснять тип орбиты (эллипс, парабола, гипербола), поскольку COMET в зависимости от заданного эксцентриситета и выбирает соответствующий метод вычисления. Вытянутые эллиптические орбиты с высоким эксцентриситетом рассчитываются посредством надежного численного метода Штумпфа.

COMET позволяет вычислять гелиоцентрические эклиптические и геоцентрические экваториальные координаты. В обоих случаях применительно к используемому равноденствию следует учитывать прецессию. При вычислении геоцентрических координат нужно дополнительно внести поправку за время распространения света. Геоцентрические координаты в такой форме (с учетом прецессии и времени распространения света) называются астрометрическими координатами. Их можно непосредственно сравнивать с положениями звезд в звездном каталоге или использовать для нахождения объекта на карте (того же равноденствия!).

При сравнении полученных результатов с другими источниками нужно убедиться, что приводимые там данные являются астрометрическими координатами, отнесенными к тому же равноденствию, а не видимыми координатами (для них дополнительно учитываются поправки за нутацию и аберрацию).

```
(*-----------------------------------------------------------*)
(*                        COMET                              *)
(*          Вычисление эфемерид комет и астероидов           *)
(*          (задача двух тел без учета возмущений            *)
(*          при любом эксцентриситете)                       *)
(*                   Версия 03 01.1989                       *)
(*-----------------------------------------------------------*)

PROGRAM COMET(INPUT,OUTPUT,COMINP);
  TYPE REAL33 = ARRAY[1..3,1..3] OF REAL;
  VAR DAY,MONTH,YEAR,NLINE                 : INTEGER;
      D,HOUR;T0,Q,ECC,TEQX0,FAC            : REAL;
      MODJD,T,T1,DT,T2,TEQX                : REAL;
```

```
      X,Y,Z,VX,VY,VZ,XS,YS,ZS                     : REAL;
      L,B,R,LS,BS,RS,RA,DEC,DELTA,DELTA0 : REAL;
      PQR,A,AS                                            : REAL33;
      COMINP                                               : TEXT;
(*-----------------------------------------------------------*)
(*   Сюда вводятся следующие подпрограммы в указанном порядке: *)
(*   SN, CS, ASN, ATN, ATN2, CART, POLAR, GMS, CUBR          *)
(*   MJD, CALDAT                                             *)
(*   ECLEQU, GAUSVEC, ORBECL                                 *)
(*   PMATECL, PRECART                                        *)
(*   ECCANOM, ELLIP, HYPANOM, HYPERB, STUMPFF, PARAB, KEPLER*)
(*   SUN200                                                  *)
(*-----------------------------------------------------------*)

(*-----------------------------------------------------------*)
(*   GETELM: Ввод элементов орбиты из файла COMINP           *)
(*-----------------------------------------------------------*)
PROCEDURE GETELM (VAR T0,Q,ECC.REAL;VAR PQR:REAL33;
                                              VAR TEQX0:REAL);
  VAR INC,LAN,AOP: REAL;
  BEGIN
    WRITELN;
    WRITELN (' COMET: Вычисление эфемерид комет и астероидов   ');
    WRITELN ('                 Версия 03.01.89                  ');
    WRITELN ('     (C) 1988 Thomas Pfleger,Oliver Montenbruck    ');
    WRITELN;
    WRITELN (' элементы орбиты из файла COMINP: ');
    WRITELN;

    (* открытие файла для чтения   *)
    RESET(COMINP);                          (* Standard Pascal     *)
(* ASSIGN(COMINP,'COMINP.DAT'); RESET(COMINP);*) (* TURBO Pascal *)
    (* RESET(COMINP,'COMINP.DAT'); *)       (* ST Pascal plus     *)

    (* чтение и вывод элементов орбиты  *)
    READLN (COMINP,YEAR,MONTH,D);
 WRITELN('момент прохожд. перигелия (Y M D)',YEAR:7,MONTH:3,D:6:2);
   READLN(COMINP,Q);
 WRITELN(' расстояние перигелия (q)      ',  Q:14:7,'    a.e.');
   READLN(COMINP,ECC);
 WRITELN(' эксцентриситет  (e)          ',  ECC:14:7);
   READLN(COMINP,INC);
 WRITELN(' наклонение орбиты (i)        ',  INC:12:5,'    град');
   READLN(COMINP,LAN);
 WRITELN(' долгота восходящего узла     ',  LAN:12:5,'    град');
```

```
  READLN(COMINP,AOP);
WRITELN(' аргумент перигелия          ',  AOP:12:5,'   град');
  READLN(COMINP,TEQX0);
WRITELN(' равноденствие               ',  TEQX0:9:2);

    DAY    := TRUNC(D); HOUR:=24.0*(D-DAY);
    T0     := ( MJD(DAY,MONTH,YEAR,HOUR) - 51544.5) / 36525.0;
    TEQX0 := (TEQX0-2000)/100;
    GAUSVEC (LAN,INC,AOP,PQR);

END;

(*-----------------------------------------------------------------*)
(* GETEPH: Ввод интервала времени для эфемерид и равноденствия     *)
(*-----------------------------------------------------------------*)
PROCEDURE GETEPH(VAR T1,DT,T2,TEQX:REAL);

  VAR YEAR,MONTH,DAY:INTEGER;
      EQX,HOUR,JD :  REAL;

  BEGIN

  WRITELN;
  WRITELN(' начало и конец эфемерид: ');
  WRITELN;
  WRITE   (' начальная дата вычислений (DD MM YYYY HH.HHH)...
  READLN (DAY,MONTH,YEAR,HOUR);
    T1 :=   ( MJD(DAY,MONTH,YEAR,HOUR) - 51544.5 ) / 36525.0;
    WRITE  (' конечная дата вычислений (DD MM YYYY HH HHH) ..
    READLN (DAY,MONTH,YEAR,HOUR);
    T2 :=   ( MJD(DAY,MONTH,YEAR,HOUR) - 51544.5 ) / 36525.0;
    WRITE  ('  шаг  (DD HH.HH)                              ...
    READLN (DAY,HOUR);
    DT :=   ( DAY + HOUR/24.0 ) / 36525.0;
    WRITELN;
    WRITE  (' желаемое равноденствие эфемерид (YYYY)        ...   ');
    READLN (EQX);
    TEQX:=  (EQX-2000.0)/100.0;

    WRITELN; WRITELN;
    WRITELN('      Дата       ET    Солнце    l         b        r',
           '              Ra       Dec                 расстояние'
    WRITELN (' ':45,'        h  m  s      o  "  "       (а.е.) '

  END;

(*-----------------------------------------------------------------*)
```

```
PROCEDURE WRTLBR(L,B,R:REAL);
  VAR H,M: INTEGER;
      S  : REAL;
  BEGIN
    GMS(L,H,M,S); WRITE  (H:5,M:3,S:5:1);
    GMS(B,H,M,S); WRITELN(H:5,M:3,TRUNC(S+0.5):3,R:11:6);
  END;
(*------------------------------------------------------------------  *)

BEGIN (* COMET *)

  GETELM (T0,Q,ECC,PQR,TEQX0); (* чтение элементов орбиты          *)
  GETEPH (T1,DT,T2,TEQX);      (* ввод интервала времени и         *)
                               (* равноденствия                    *)
  NLINE := 0;

  PMATECL (TEQX0,TEQX,A); (* вычисление прецессионной матрицы      *)

  T := T1;

  REPEAT

    (* дата  *)

  MODJD:= T*36525.0+51544.5; CALDAT (MODJD,DAY,MONTH,YEAR,HOUR);

    (* эклиптические координаты Солнца, равноденствие TEQX         *)

    SUN200 (T,LS,BS,RS);     CART    (RS,BS,LS,XS,YS,ZS);
    PMATECL (T,TEQX,AS);     PRECART  (AS,XS,YS,ZS),

    (* гелиоцентрические эклиптические координаты кометы           *)

    KEPLER  (T0,T,Q,ECC,PQR,X,Y,Z,VX,VY,VZ);
  PRECART (A,X,Y,Z);    PRECART  (A,VX,VY,VZ);  POLAR  (X,Y,Z,R,B,L);

    (* геометрические геоцентрические координаты кометы            *)

    X:=X+XS; Y:=Y+YS; Z:=Z+ZS; DELTA0:= SQRT ( X*X + Y*Y + Z*Z );

    (* поправка первого порядка за время распространения света     *)

    FAC:=0.00578*DELTA0;  X:=X-FAC*VX; Y:=Y-FAC*VY; Z:=Z-FAC*VZ;
    ECLEQU (TEQX,X,Y,Z);  POLAR (X,Y,Z,DELTA,DEC,RA); RA:=RA/15.0;

    (* вывод    *)
```

```
    WRITE(DAY:3,'.',MONTH:2,'.',YEAR:4,HOUR:5:1);
    WRITE(LS:7:1,L:7:1,B:6:1,R:7:3);      WRTLBR(RA,DEC,DELTA0);
    NLINE:= NLINE+1; IF (NLINE MOD 5) = 0 THEN WRITELN;

    (* следующий момент времени                               *)
    T := T + DT;

  UNTIL (T2<T);

END.
(*-----------------------------------------------------------*)
```

В качестве примера использования программы COMET попробуем вычислить эфемериды кометы Галлея во время ее последнего приближения к Земле. Для этого нужно предварительно заготовить файл с именем COMINP.DAT с элементами орбиты. Комментарии (восклицательный знак и следующее за ним разъяснение) нужны только для информации и по желанию могут быть пропущены. Для кометы Галлея 1985/1986 гг. данные будут выглядеть так:

```
1986 2 9.43867  ! время прохождения перигелия (год месяц день.доля)
   0.5870992    ! расстояние перигелия q а.е.
   0.9672725    ! эксцентриситет е
 162.23932      ! наклонение орбиты i
  58.14397      ! долгота восходящего узла
 111.84658      ! аргумент перигелия
1950.0          ! равноденствие элементов орбиты
```

В начале работы, прежде чем пользователь задаст временной интервал, шаг и желаемое равноденствие, программа COMET в качестве контроля выдает считанные из файла COMINP.DAT элементы орбиты:

```
        COMET: Вычисление эфемерид комет и астероидов
                       Версия 03.01.89
          (C) 1988 Thomas Pfleger, Oliver Montenbruck

  Элементы орбиты из файла COMINP:
  момент прохожд. перигелия (Y M D)       1986 2 9.44
  расстояние перигелия  (q)                  0.5870992 а.е.
  эксцентриситет        (e)                  0.9672725
  наклонение орбиты     (i)                162.23932 град
  долгота восходящего узла                  58.14397 град
  аргумент перигелия                       111.84658 град
  равноденствие                           1950.00
```

Мы хотим получить эфемериды для периода от 15 октября 1985 г. до 1 апреля 1986 г. с шагом 10 сут на равноденствие 2000 г. Граничные даты эфемерид задаются в виде «день, месяц, год и час с десятичными долями». Значение шага вводится в форме «день и час с десятичными долями». Ниже вводимые данные набраны курсивом.

Начало и конец эфемерид:
начальная дата вычислений (DD MM YYYY HH.HHH)... *15 11 1985 0.0*
конечная дата вычислений (DD MM YYYY HH.HHH)... *1 04 1986 0.0*
шаг (DD HH.HH) ... *10 00.0*
желаемое равноденствие эфемерид (YYYY) ... *2000.0*

По заданным величинам COMET вычислит следующие эфемериды:

Дата	ET	Солнце	l	b	r	Ra h	m	s	Dec °	′	″	Расстояние (а.е.)
15.11.1985	0.0	232.6	56.9	0.6	1.720	4	0	40.3	22	4	28	0.736822
25.11.1985	0.0	242.7	53.2	1.8	1.572	2	15	19.1	18	24	12	0.623053
5.12.1985	0.0	252.8	48.7	3.2	1.422	0	28	47.5	10	39	0	0.665665
15.12.1985	0.0	263.0	43.1	5.0	1.269	23	18	18.6	3	53	22	0.821652
25.12.1985	0.0	273.1	35.9	7.1	1.114	22	36	56.1	0	−20	5	1.019722
4. 1.1986	0.0	283.3	26.2	9.8	0.961	22	10	36.0	-3	0	26	1.216950
14. 1.1986	0.0	293.5	12.8	13.0	0.814	21	50	58.9	-4	58	44	1.388260
24. 1.1986	0.0	303.7	353.4	16.2	0.687	21	33	26.2	-6	47	36	1.511970
3. 2.1986	0.0	313.9	326.3	17.7	0.604	21	15	33.4	-8	49	26	1.563479
13. 2.1986	0.0	324.0	294.8	14.9	0.592	20	57	11.7	-11	15	58	1.520206
23. 2.1986	0.0	334.1	267.2	8.7	0.658	20	39	33.7	-14	9	18	1.383203
5. 3.1986	0.0	344.1	247.1	2.6	0.775	20	22	12.2	-17	39	45	1.179038
15. 3.1986	0.0	354.1	232.8	-1.9	0.918	20	0	46.1	-22	28	16	0.936978
25. 3.1986	0.0	4.1	222.5	-5.2	1.070	19	22	34.1	-30	13	41	0.685468

Первые два столбца содержат дату и время. Далее следует столбец с геоцентрической эклиптической долготой Солнца. С помощью этих данных можно оценить, насколько далеки комета или астероид от Солнца. Отсюда следует указание на то, доступен ли объект для наблюдений. Три последующих столбца (l, b, r) содержат гелиоцентрические эклиптические координаты небесного тела. В последних трех столбцах даны геоцентрические координаты — прямое восхождение, склонение и геоцентрическое геометрическое (без учета времени распространения света) расстояние кометы в астрономических единицах.

5. ОРБИТЫ ПЛАНЕТ

Если не принимать во внимание Меркурий и Плутон, то можно заметить важную общую черту больших планет — малый эксцентриситет орбит и малый наклон орбит к эклиптике В отличие от планет орбиты комет расположены в пространстве беспорядочно и имеют разнообразные формы Кроме того, близкие к круговым орбиты планет сильно удалены друг от друга и никогда близко не сходятся

Для вычисления эфемерид эти свойства орбит планет имеют два важных следствия Первое касается определения кеплеровской орбиты без учета возмущений, описанного в гл 4 для случая орбиты произвольной формы Из за небольшого эксцентриситета истинная аномалия испытывает лишь небольшие периодические уклонения от средней аномалии Ее легче найти посредством разложения в ряд, чем решением уравнения Кеплера То же самое касается определения расстояния до Солнца, которое меняется всего на несколько процентов от значения большой полуоси Кроме того, можно в простой приближенной форме представить отклонение орбиты от эклиптики, что позволяет отказаться от трудоемких вычислений векторов Гаусса Эти приближения подробнее будут описаны в следующем разделе

Второе следствие касается изменений планетных орбит под действием взаимного тяготения Так как планеты не приближаются друг к другу, эти силы во много раз меньше силы притяжения Солнца Значительных изменений орбиты, которые случаются с кометой при прохождении мимо Юпитера или Сатурна, в движении планет не наблюдается Поэтому орбиты планет могут быть описаны средними кеплеровскими эллипсами, на которые налагаются небольшие периодические возмущения

Эфемериды планет сегодня обычно вычисляют с помощью численных итерационных методов и после этого записывают на магнитную ленту Благодаря возможностям современных вычислительных машин, эти методы отличаются простотой и точностью Однако им свойствен и недостаток в отличие от случая чистой задачи двух тел аналитическое представление движения планет здесь отсутствует Большим достижением небесной механики стало создание метода разложения в ряд, позволяющего вычислить отклонение движения планеты от невозмущенной орбиты для любого момента времени Уже по длине ряда можно судить, насколько сложна и трудоемка работа его составления Для нашей задачи — точного вычисления положений планет — достаточно понимать общую структуру разложения в ряд и уметь проводить его численный расчет Детальнее с разложением возмущений в ряды можно ознакомиться в учебниках небесной механики

Среди аналитических представлений движения планет наиболее известны таблицы движения внутренних планет от Меркурия до Марса, составленные Саймоном Ньюкомом в Морской обсерватории США Хотя эти таблицы были вычислены еще в конце прошлого века, вплоть до самого недавнего времени они служили основой соответствующих эфемерид *Астрономического альманаха* (*Astronomical Almanac*) Подобные же таблицы для внешних планет составили Дж Хилл для Юпитера и Сатурна и С Ньюком для Урана и Нептуна Однако вскоре их сменили первые численно рассчитанные эфемериды, поскольку на больших промежутках времени точность этих таблиц была ниже, чем для внутренних планет

Настоящая глава написана на основе названных выше работ и разложения в ряд орбиты Плутона, полученного Э Гоффином, Дж Меесом и К Стейартом Возмущающие члены в выражениях для разных планет различаются иногда весьма существенно, так что не удается единообразно перевести их на язык программы По этой причине пришлось вначале переработать все ряды с помощью программы преобразования формул В результате кеплеровское движение и возмущающие члены, независимо от первоначального представления, были выражены в виде периодического движения по долготе, широте и расстоянию Кроме того, было отброшено множество членов, которые из за своей незначительности (как правило, << 1″) не влияют на наблюдаемую точность ряда Это преобразование позволяет создать сравнительно короткую и компактную программу, не ухудшая точности разложения в ряд

5.1. Использование метода разложения в ряд для решения задачи Кеплера

Путь планеты по невозмущенному эллипсу Кеплера является строго периодической функцией средней аномалии M При изменении средней аномалии на 360° планета, сделав полный оборот, оказывается на том же месте Поэтому каждая функция положения на орбите может быть представлена в виде ряда Фурье, т е суммой синусов и косинусов аргументов, кратных M Примером служат разложения в ряд разности истинной и средней аномалий

$$v - M = a_1 \sin(M) + a_1 \sin(2M) + a_1 \sin(3M) +$$

и расстояния r в единицах большой полуоси a

$$r/a = b_0 + b_1 \cos(M) + b_2 \cos(2M) +$$

Коэффициенты a_i и b_i в этих рядах являются функциями эксцентриситета орбиты e Главными в разложении $v - M$ (так называемое *уравнение центра*) и r/a являются члены до второй степени e

$$v - M = 2e \sin(M) + \frac{5}{4} e^2 \sin(2M) + \ldots \quad (\text{рад}) \qquad (5\ 1)$$

и

$$r/a = \left[1 + \frac{1}{2} e^2\right] - e \cos(M) - \frac{1}{2} e^2 \cos(M) + \ldots \qquad (5\ 2)$$

При малых эксцентриситетах оба ряда очень быстро сходятся, так как последовательные члены уменьшаются примерно в e раз. Для орбиты Земли с $e = 0{,}0167$ получим

$$\begin{aligned} \nu &= M + 0{,}0334 \sin(M) + 0{,}00035 \sin(2M) + \dots \text{ (рад.)}, \\ &= M + 1{,}916^\circ \sin(M) + 0{,}020^\circ \sin(2M) + \dots, \end{aligned}$$

$$r = a\,(1{,}00014 - 0{,}0167 \cos(M) - 0{,}00014 \cos(2M) + \dots).$$

Ошибка этой короткой формулы составляет только $1''$ или 10^{-5} а.е. и может быть еще уменьшена добавлением дополнительных членов.

Для вывода вышеприведенных формул нужны более глубокие знания, а мы выведем только первый член уравнения центра. Вспомним уравнение Кеплера (4.7)

$$E - e \sin E = M$$

Так как E и M отличаются друг от друга одним членом порядка e, $e \sin E$ можно заменить на $e \sin M$, в результате для разности между эксцентрической и истинной аномалиями получим

$$E - M \approx e \sin M.$$

Далее, из уравнения конического сечения $r = a\,(1 - e^2)/(1 + e\cos\nu)$ при $r\cos\nu = a(\cos E - e)$ следует зависимость

$$\cos\nu \approx \cos E - e\,(1 - \cos\nu \cos E).$$

Во втором члене правой части $\cos\nu$ можно заменить на $\cos E$, откуда

$$\cos\nu \approx \cos E - \{e \sin E\} \sin E$$

Сравнив это соотношение с разложением в ряд Тейлора функции косинуса

$$\cos(x + \Delta x) \approx \cos x - \Delta x \sin x,$$

видим, что при малых эксцентриситетах разность $\nu - E$ приблизительно равна $e \sin E$ Поэтому первый член уравнения центра, как и указывалось выше, будет

$$\nu - M = (\nu - E) + (E - M) \approx e \sin E + e \sin M \approx 2e \sin M.$$

При небольшом наклоне i орбита всюду проходит вблизи эклиптики При этом эклиптическая долгота положения планеты отличается от суммы $\bar{\omega} + \nu$ долготы перигелия и истинной аномалии на малую величину R

(*приведение к эклиптике*), которая равна нулю при $i = 0$, а при небольших наклонах орбиты имеет в первом приближении значение

$$R = 1 - (\nu + \bar{\omega}) \approx - \operatorname{tg}^2\left[\frac{i}{2}\right] \sin(2u) \quad \text{(рад.)} \tag{5.3}$$

при $u = \nu + \omega = \nu + \bar{\omega} - \Omega$. Так как $\nu \approx M$ ($u \approx M + \omega$), можно R записать так:

$$R \approx - \operatorname{tg}^2\left[\frac{i}{2}\right]\left\{\sin(2\omega)\cos(2M) + \cos(2\omega)\sin(2M)\right\}$$

Умножая это значение на $180°/\pi$, получаем R в градусах. Уравнение 4.20)

$$z = r\,(\sin(\nu + \omega)\sin(i)) \tag{5.4}$$

для z-координаты как функции элементов орбиты приводит с учетом соотношения

$$\sin(b) = z/r \tag{5.5}$$

при малых наклоне орбиты i и эклиптической широте b к следующему приближению:

$$b \approx i\sin(\nu + \omega) = \{i\sin\omega\}\cos(\nu) + \{i\cos\omega\}\sin(\nu)$$

Заменяя ν через $\nu \approx M + 2e\sin M$ и разлагая в ряд Тейлора, получаем

$$\cos\nu \approx \cos M - (2e\sin M)\sin M = \cos(M) - e\left[1 - \cos(M)\right],$$

$$\sin\nu \approx \sin M + (2e\sin M)\cos M = \sin(M) + e\sin(2M),$$

поэтому

$$\begin{aligned} b \approx\ &\{i\sin\omega\}\{\cos(M) - e\,(1 - \cos(2M))\} + \\ &+ \{i\cos\omega\}\{\sin(M) + e\sin(2M)\}\,. \end{aligned}$$

При представлении орбиты планеты без учета возмущений в виде тригонометрических рядов, основными членами будут:

$$\begin{aligned} l =\ &\bar{\omega} + M + \\ &+ \frac{180°}{\pi}\{2e\}\sin(M) + \\ &+ \frac{180°}{\pi}\left\{\frac{5}{4}e^2 - \operatorname{tg}^2\left[\frac{i}{2}\right]\cos(2\omega)\right\}\sin(2M) + \\ &+ \frac{180°}{\pi}\left\{-\operatorname{tg}^2\left[\frac{i}{2}\right]\sin(2\omega)\right\}\cos(2M), \end{aligned} \tag{5.6}$$

$$b = -\{\iota e \sin \omega\} +$$

$$+ \{\iota \sin \omega\} \cos (M) + \{\iota \cos \omega\} \sin (M) +$$

$$+ \{\iota e \sin \omega\} \cos (2M) + \{\iota e \cos \omega\} \sin (2M), \quad (5\ 7)$$

$$r = \{a(1 + e^2/2)\} -\{ae\} \cos (M) - \{ae^2/2\} \cos (2M) \quad (5\ 8)$$

Подставляя в эти уравнения численные значения элементов орбиты получают компактные формулы для эклиптических координат (l, b, r) в зависимости от средней аномалии M Для Юпитера, например, при

$$a = 5{,}2 \text{ а е}, \quad \Omega = 100{,}0°, \quad \omega = 274{,}0°,$$

$$e = 0{,}048, \quad \iota = 1{,}31°, \quad \bar{\omega} = 14{,}0°$$

получим

$$l = M + 14{,}0° + 19800'' \sin (M) + 620'' \sin (2M) + 4'' \cos (2M),$$

$$b = +226'' - 4700'' \cos (M) + 329'' \sin (M) -$$

$$-226'' \cos (2M) + 16'' \sin (2M),$$

$$r = (5{,}206 - 0{,}250 \cos (M) - 0{,}006 \cos (2M)) \text{ а е}$$

5.2. Возмущающие члены

Взаимные возмущения планет, вызывающие отклонения от кеплеровской орбиты, проявляются в дополнительных членах разложения, содержащих наряду с аномалиями возмущаемых планет также аномалии возмущающих планет В дальнейшем M_n будет обозначать среднюю аномалию n й планеты, например M_5 и M_6—средние аномалии Юпитера и Сатурна

$M_1 = 0^r{,}4855407 + 415^r{,}2014314\ T$ (Меркурий),

$M_2 = 0^r{,}1400197 + 162^r{,}5494552\ T$ (Венера),

$M_3 = 0^r{,}9931266 + 99^r{,}9973604\ T$ (Солнце, Земля),

$M_4 = 0^r{,}0538553 + 53^r{,}1662736\ T$ (Марс),

$$M_5 = 0{,}^r0565314 + 8{,}^r4302963\ T \quad \text{(Юпитер)},$$

$$M_6 = 0{,}^r8829867 + 3{,}^r3947688\ T \quad \text{(Сатурн)}, \qquad (5.9)$$

$$M_7 = 0{,}^r3967117 + 1{,}^r1902849\ T \quad \text{(Уран)},$$

$$M_8 = 0{,}^r72149906 + 0{,}^r6068526\ T \quad \text{(Нептун)},$$

$$M_9 = 0{,}^r0385795 + 0{,}^r4026667\ T \quad \text{(Плутон)}$$

Значения здесь приведены в долях орбитального оборота (1^r)[1] Умножив их на 360° или 2π, получим среднюю аномалию в привычном градусном или радианном измерении Время

$$T = (\mathrm{JD} - 2451545)/36525,$$

как обычно, исчисляется в юлианских столетиях с J2000

Различные возмущающие члены для ясности можно представить в виде таблицы, например в табл 5 1 для Юпитера В таблице содержатся периодические вариации эклиптической долготы (dl), широты (db) и расстояния от Солнца (dr) Члены с равными аргументами тригонометрических функций собраны в одной строке, что облегчает пользование таблицей Например, взяв значения из строки, отмеченной звездочкой, получим члены, которые нужно добавить к долготе, широте и расстоянию

$$dl = \quad 68{,}5'' \cos(3M_5 - 5M_6) + 134{,}3'' \sin(3M_5 - 5M_6),$$

$$dr = (-166{,}9 \quad \cos(3M_5 - 5M_6) + 86{,}5 \quad \sin(3M_5 - 5M_6)\ 10^{-6}\ \text{а е},$$

$$db = \quad 7{,}1'' \cos(3M_5 - 5M_6) + 15{,}2'' \sin(3M_5 - 5M_6)$$

Члены, умножаемые на время T или на T^2, обозначены соответственно 1 или 2 в столбце ι_T

---

1) Некоторые используемые в программе значения немного отличаются от приведенных в тексте Это не ошибка, а следствие применения разных теорий разложения в ряд.

Таблица 5.1. Периодические члены орбиты Юпитера

i_5		i_T	dl [″] cos	dl [″] sin	dr [10^{-5} а.е.] cos	dr [10^{-5} а.е.] sin	db [″] cos	db [″] sin
				Кеплеровские члены				
1		0	−113,1	19998,6	−25208,2	−142,2	−4670,7	288,9
1		1	−76,1	66,9	−84,2	−95,8	21,6	29,4
1		2	−0,5	−0,3	0,4	−0,7	0,1	−0,1
2		0	−3,4	632,0	−610,6	−6,5	−226,8	12,7
2		1	−4,2	3,8	−4,1	−4,5	0,2	0,6
3		0	−0,1	28,0	−22,1	−0,2	−12,5	0,7
4		0	0,0	1,4	−1,0	0,0	−0,6	0,0
i_5	i_6	i_T						
				Сатурн				
−1	−1	0	−0,2	1,4	2,0	0,6	0,1	−0,2
0	−1	0	9,4	8,9	3,9	−8,3	−0,4	−1,4
0	−2	0	5,6	−3,0	−5,4	−5,7	−2,0	0,0
0	−3	0	−4,0	−0,1	0,0	5,5	0,0	0,0
0	−5	0	3,3	−1,6	−1,6	−3,1	−0,5	−1,2
1	−1	0	78,8	−14,5	11,5	64,4	−0,2	0,2
1	−2	0	−2,0	−132,4	28,8	4,3	−1,7	0,4
1	−2	1	−1,1	−0,7	0,2	−0,3	0,0	0,0
1	−3	0	−7,5	−6,8	−0,4	−1,1	0,6	−0,9
1	−4	0	0,7	0,7	0,6	−1,1	0,0	−0,2
1	−5	0	51.5	−26,0	−32,5	−64,4	−4,9	−12,4
1	−5	1	−1,2	−2,2	−2,7	1,5	−0,4	0,3
2	−1	0	5,3	−0,7	0,7	6,1	0,2	1,1
2	−2	0	−76,4	−185,1	260,2	−108,0	1,6	0,0
2	−3	0	66,7	47,8	−51,4	69,8	0,9	0,3
2	−3	1	0,6	−1,0	1,0	0,6	0,0	0,0
2	−4	0	17,0	1,4	−1,8	9,6	0,0	−0,1
2	−5	0	1066,2	−518,3	−1,3	−23,9	1,8	−0,3
2	−5	1	−25,4	−40,3	−0,9	0,3	0,0	0,0
2	−5	2	−0,7	0,5	0,0	0,0	0,0	0,0
3	−2	0	−5,0	−11,5	11,7	−5,4	2,1	−1,0
3	−3	0	16,9	−6,4	13,4	26,9	−0,5	0,8
3	−4	0	7,2	−13,3	20,9	10,5	0,1	−0,1
3	−5	0	68,5	134,3	−166,9	86,5	7,1	15,2*
3	−5	1	3,5	−2,7	3,4	4,3	0,5	−0,4
3	−6	0	0,6	1,0	−0,9	0,5	0,0	0,0
3	−7	0	−1,1	1,7	−0,4	−0,2	0,0	0,0
4	−2	0	−0,3	−0,7	0,4	−0,2	0,2	−0,1
4	−3	0	1,1	−0,6	0,9	1,2	0,1	0,2
4	−4	0	3,2	1,7	−4,1	5,8	0,2	0,1
4	−5	0	6,7	8,7	−9,3	8,7	−1,1	1,6
4	−6	0	1,5	−0,3	0,6	2,4	0,0	0,0
4	−7	0	−1,9	2,3	−3,2	−2,7	0,0	−0,1
4	−8	0	0,4	−1,8	1,9	0,5	0,0	0,0
4	−9	0	−0,2	−0,5	0,3	−0,1	0,0	0,0
4	−10	0	−8,6	−6,8	−0,4	0,1	0,0	0,0
4	−10	1	−0,5	0,6	0,0	0,0	0,0	0,0
5	−5	0	−0,1	1,5	−2,5	−0,8	−0,1	0,1
5	−6	0	0,1	0,8	−1,6	0,1	0,0	0,0
5	−9	0	−0,5	−0,1	0,1	−0,8	0,0	0,0
5	−10	0	2,5	−2,2	2,8	3,1	0,1	−0,2
i_5	i_7	i_T						
				Уран				
1	−1	0	0,4	0,9	0,0	0,0	0,0	0,0
1	−2	0	0,4	0,4	−0,4	0,3	0,0	0,0
i_5	i_7	i_T						
				Сатурн и Уран				
2	−6	3	−0,8	8,5	−0,1	0,0	0,0	0,0
3	−6	3	0,4	0,5	−0,7	0,5	−0,1	0,0

Влияние других планет проявляется не только в периодических возмущениях, но и в вековых изменениях средних элементов орбиты. В результате в членах, описывающих само кеплеровское движение, возникают составляющие, зависящие от времени T. Продемонстрируем это вновь на примере орбиты Юпитера. В подпрограмме JUP200 используются следующие уравнения для вычисления координат Юпитера:

$$l = M_5 + 14^\circ\!,00076 + (5025''\!,2 + 0''\!,8\ T)T + dl,$$

$$b = 227''\!,3 - 0''\!,3\ T + db,$$

$$r = (5{,}208873 + 0{,}0000417)\ \text{а.е.} + dr.$$

Здесь dl, db и dr содержат все периодические составляющие орбитального движения, т. е. кеплеровские члены и возмущения, вызванные Сатурном и Ураном. Определенные таким образом значения эклиптической долготы и широты относятся к *среднему равноденствию даты*. Формула для l содержит большой вековой член $+5025''\!,2\ T$, описывающий движение точки весеннего равноденствия.

Из-за недостатка места мы не можем рассмотреть все возмущающие члены для всех планет. Однако благодаря единообразию структур разложения в ряд объяснений, относящихся к орбите Юпитера, должно быть достаточно для понимания формул и членов разложения, используемых в программе.

Правда, расчетам, относящимся к орбите Земли, свойственны некоторые особенности. Земля вращается вокруг Солнца вместе с Луной, и обе массы можно объединить в одну, расположенную в их общем центре. Движение этого центра определяет среднюю эклиптику и его, как и для других планет, можно описать в виде возмущенной кеплеровской орбиты и представить разложением в ряд. Как следует из соотношения масс Луны и Земли, расстояние центра Земли от общего центра масс составляет 1/81 расстояния Луны от Земли, что равно 3/4 радиуса Земли (рис. 5.1). Если смотреть с Солнца, эклиптическая долгота Земли колеблется около среднего положения с амплитудой 7″ в течение месяца. Эклиптическая широта Земли за это время меняется на 0,5″.

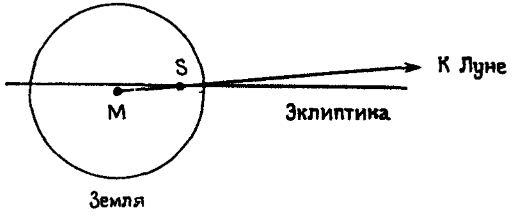


Рис. 5.1. Центр масс системы Земля—Луна.

5.3. Численные расчеты разложений в ряд

Чтобы обеспечить точность в несколько секунд дуги, нужно для каждой планеты использовать сотни возмущающих членов. И если все это в явном виде запрограммировать, получим длинный и неудобный текст программы, который к тому же потребует излишне длительного счета. Большое время, необходимое для вычислений, обусловлено при этом использованием тригонометрических функций, которые по сравнению с элементарными операциями сложения и вычитания вычисляются медленно. Задача ускорения счета тем более актуальна даже при наличии быстродействующих компьютеров, поскольку в разложении у нас используется очень много угловых функций.

Можно существенно облегчить работу, используя теоремы сложения для синуса и косинуса. Они позволяют по известным функциям отдельных углов вычислить угловые функции суммы или разности:

$$\begin{aligned} \cos(\alpha_1+\alpha_2) &= \cos\alpha_1\cos\alpha_2 - \sin\alpha_1\sin\alpha_2, \\ \sin(\alpha_1+\alpha_2) &= \sin\alpha_1\cos\alpha_2 + \cos\alpha_1\sin\alpha_2. \end{aligned} \tag{5.10}$$

При этом очень удобно на основе значений cos M и sin M вычислить и запомнить значения cos (iM) и sin (iM) для углов, кратных средней аномалии планеты M, при i = 1,2,3... . Это соображение можно проиллюстрировать небольшим примером:

```
VAR C,S:  ARRAY[0..5] OF REAL; (* место для значений cos(iM),   *)
                                        (*  sin(iM)            *)
    M  : REAL;
    I  : INTEGER;
PROCEDURE ADDTHE(C1,S1,C2,S2:REAL; VAR C,S:REAL);
 BEGIN
  C    := C1*C2 - S1*S2; (* cos(a1+a2)=cos(a1)cos(a2)          *)
                                   (* - sin(a1)sin(a2)         *)
  S    := S1*C2 + C1*S2; (*sin(a1+a2)= sin(a1)cos(a2)
                                   (* + cos(a1)sin(a2)         *)
 END;
 BEGIN
  WRITE('M (рад) ?'); READELN(M); (* ввод средней аномалии M    *)
  C[0] :=1.0;    C[1]  := COS(M);  (* cos(0*M), cos(1*M)        *)
  S[0] :=0.0;  S[1]    := SIN(M);  (* sin(0*M), sin(1*M)        *)
  FOR I =2 TO 5 DO   (* вычисление cos(iM), sin(iM), i=2..5     *)
    ADDTHE ( C[1],S[1], C[I-1],S[I-1], C[I],S[I] );
     FOR I :=0 TO 5 DO          (* вывод данных                 *)
       WRITELN (I:5, C[I]:12:7, S[I]:10:7 );
 END;
```

Следующий пример процедуры JUP200 показывает, как вычисляют и суммируют возмущающие члены по этой схеме. Подпрограмма TERM вычисляет для индексов i_5 и i_j значения $u = \cos(i_5 M_5 + i_j M_j)$ и $v = \sin(i_5 M_5 + i_j M_j)$, умножает их на соответствующий коэффициент и суммирует с dl, dr и db. Если все члены с одинаковыми i_5 и i_j и разными $i_T = 0,1...$ определяются последовательно, один за другим, достаточно, как это сделано в подпрограмме TERM, вычислить u и v при $i_T = 0$, а значения для $i_T = 1$ или $i_T = 2$ получить умножением на T. Некоторая громоздкость такого подхода быстро окупится, если, как в нашем случае, используется очень большое число членов.

```
(*-----------------------------------------------------------------  *)
(* JUP200:  Юпитер; эклиптические координаты L,B,R (град. и а е.    *)
(*          Равноденствие даты                                      *)
(*          (T: время в юлианских столетиях с J2000)                *)
(*          (   = (JED-2451545.0)/36525               )             *)
(*-----------------------------------------------------------------  *)
PROCEDURE JUP200(T:REAL;VAR L,B,R:REAL);
  CONST P2=6.283185307;
  VAR C5,S5        : ARRAY [-1..5]   OF REAL;
     C,S           : ARRAY [-10..0] OF REAL;
     M5,M6,M7      : REAL;
     U,V,DL,DR,DB: REAL;
     I             : INTEGER;

  FUNCTION FRAC(X:REAL):REAL;
   BEGIN X:=X-TRUNC(X); IF (X<0) THEN X:=X+1.0; FRAC:=X  END;

  PROCEDURE ADDTHE(C1,S1,C2,S2:REAL; VAR C,S:REAL);
   BEGIN C:=C1*C2-S1*S2; S:=S1*C2+C1*S2; END;

  PROCEDURE TERM(I5,I,IT:INTEGER;DLC,DLS,DRC,DRS,DBC,DBS:REAL);
   BEGIN
    IF  IT=0  THEN  ADDTHE(C5[I5],S5[I5],C[I],S[I],U,V)
                ELSE BEGIN U:=U*T; V:=V*T END;
    DL:=DL+DLC*U+DLS*V; DR:=DR+DRC*U+DRS*V; DB:=DB+DBC*U+DBS*V;
   END;

  PROCEDURE PERTSAT;(* Кеплеровские члены и возмущения от        *)
                    (*             Сатурна                       *)
  VAR I: INTEGER;
  BEGIN
    C[0]:=1.0;   S[0]:=0.0;   C[-1]:=COS(M6);    S[-1]:=-SIN(M6);
FOR I:=-1 DOWNTO -9 DO ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);

   TERM(-1,  -1,0,   -0.2,      1.4,       2.0,       0.6,       0.1,    -0.2);
   TERM( 0,  -1,0,    9.4,      8.9,       3.9,      -8.3,      -0.4,    -1.4);
   TERM( 0,  -2,0,    5.6,     -3.0,      -5.4,      -5.7,      -2.0,     0.0);
   TERM( 0,  -3,0,   -4.0,     -0.1,       0.0,       5.5,       0.0,     0.0);
```

```
  TERM( 0,  -5,0,     3.3,     -1.6,      -1.6,    -3.1,    -0.5,  -1.2);
  TERM( 1,   0,0,  -113.1, 1 998.6,  -25208.2,  -142.2, -4670.7, 288.9);
  TERM( 1,   0,1,   -76.1,     66.9,     -84.2,   -95.8,    21.6,  29.4);
  TERM( 1,   0,2,    -0.5,     -0.3,       0.4,    -0.7,     0.1,  -0.1);
  TERM( 1,  -1,0,    78.8,    -14.5,      11.5,    64.4,    -0.2,   0.2);
  TERM( 1,  -2,0,    -2.0,   -132.4,      28.8,     4.3,    -1.7,   0.4);
  TERM( 1,  -2,1,    -1.1,     -0.7,       0.2,    -0.3,     0.0,   0.0);
  TERM( 1,  -3,0,    -7.5,     -6.8,      -0.4,    -1.1,     0.6,  -0.9);
  TERM( 1,  -4,0,     0.7,      0.7,       0.6,    -1.1,     0.0,  -0.2);
  TERM( 1,  -5,0,    51.5,    -26.0,     -32.5,   -64.4,    -4.9, -12.4);
  TERM( 1,  -5,1,    -1.2,     -2.2,      -2.7,     1.5,    -0.4,   0.3);
  TERM( 2,   0,0,    -3.4,    632.0,    -610.6,    -6.5,  -226.8,  12.7);
  TERM( 2,   0,1,    -4.2,      3.8,      -4.1,    -4.5,     0.2,   0.6);
  TERM( 2,  -1,0,     5.3,     -0.7,       0.7,     6.1,     0.2,   1.1);
  TERM( 2,  -2,0,   -76.4,   -185.1,     260.2,  -108.0,     1.6,   0.0);
  TERM( 2,  -3,0,    66.7,     47 8,     -51.4,    69.8,     0.9,   0.3);
  TERM( 2,  -3,1,     0.6,     -1.0,       1.0,     0.6,     0.0,   0.0);
  TERM( 2,  -4,0,    17.0,      1 4,      -1.8,     9.6,     0.0,  -0.1);
  TERM( 2,  -5,0,  1066.2,   -518.3,      -1.3,   -23.9,     1.8,  -0.3);
  TERM( 2,  -5,1,   -25.4,    -40.3,      -0.9,     0.3,     0.0,   0.0);
  TERM( 2,  -5,2,    -0.7,      0.5,       0.0,     0.0,     0.0,   0.0);
  TERM( 3,   0,0,    -0.1,     28.0,     -22.1,    -0.2,   -12.5,   0.7);
  TERM( 3,  -2,0,    -5.0,    -11.5,      11.7,    -5.4,     2.1,  -1.0);
  TERM( 3,  -3,0,    16.9,     -6.4,      13.4,    26.9,    -0.5,   0.8);
  TERM( 3,  -4,0,     7.2,    -13.3,      20.9,    10.5,     0.1,  -0.1);
  TERM( 3,  -5,0,    68.5,    134.3,    -166.9,    86.5,     7.1,  15.2);
  TERM( 3,  -5,1,     3.5,     -2.7,       3.4,     4.3,     0.5,  -0.4);
  TERM( 3,  -6,0,     0.6,      1.0,      -0.9,     0.5,     0.0,   0.0);
  TERM( 3,  -7,0,    -1.1,      1.7,      -0.4,    -0.2,     0.0,   0.0);
  TERM( 4,   0,0,     0.0,      1.4,      -1.0,     0.0,    -0.6,   0.0);
  TERM( 4,  -2,0,    -0.3,     -0.7,       0.4,    -0.2,     0.2,  -0.1);
  TERM( 4,  -3,0,     1.1,     -0.6,       0.9,     1.2,     0.1,   0.2);
  TERM( 4,  -4,0,     3.2,      1.7,      -4.1,     5.8,     0.2,   0.1);
  TERM( 4,  -5,0,     6.7,      8.7,      -9.3,     8.7,    -1.1,   1.6);
  TERM( 4,  -6,0,     1.5,     -0.3,       0.6,     2.4,     0.0,   0.0);
  TERM( 4,  -7,0,    -1.9,      2.3,      -3.2,    -2.7,     0.0,  -0.1);
  TERM( 4,  -8,0,     0.4,     -1.8,       1.9,     0.5,     0.0,   0.0);
  TERM(4,   -9,0,    -0.2,     -0.5,       0.3,    -0.1,     0.0,   0.0);
  TERM( 4,-10,0,     -8.6,     -6.8,      -0.4,     0.1,     0.0,   0,0);
  TERM( 4,-10,1,     -0.5,      0.6,       0.0,     0.0,     0.0,   0.0);
  TERM( 5,  -5,0,    -0.1,      1.5,      -2.5,    -0.8,    -0.1,   0.1);
  TERM( 5,  -6,0,     0.1,      0.8,      -1.6,     0.1,     0.0,   0.0);
  TERM( 5,  -9,0,    -0.5,     -0.1,       0.1,    -0.8,     0.0,   0.0);
  TERM( 5,-10,0,      2.5,     -2.2,       2.8,     3.1,     0.1,  -0.2);
END;
PROCEDURE PERTURA;(* Возмущения от Урана   *)
```

```
  BEGIN
   C[-1]:=COS(M7);       S[-1]:=-SIN(M7);
   ADDTHE(C[-1],S[-1],C[-1],S[-1],C[-2],S[-2]);
   TERM( 1,  -1,   0,    0.4,      0.9,      0.0,      0.0,       0.0,     0.0);
   TERM( 1,  -2,   0,    0.4,      0.4,     -0.4,      0.3,       0.0,     0.0);
  END;

PROCEDURE PERTSUR;(* Возмущения от Сатурна и Урана   *)
   VAR PHI,X,Y: REAL;
   BEGIN
    PHI:=(2*M5-6*M6+3*M7);  X:=COS(PHI),   Y:=SIN(PHI);
    DL :=DL-0.8*X+8 5*Y;     DR:=DR-0.1*X;
    ADDTHE(X,Y,C5[1],S5[1],X,Y);
    DL :=DL+0.4*X+0.5*Y;    DR:=DR-0.7*X+0.5*Y;   DB:=DB-0.1*X;
   END;

  BEGIN   (* JUP200    *)

  DL    :=0.0;    DR:=0.0;    DB:=0.0;
  M5    :=P2*FRAC(0.0565314+8.4302963*T);
  M6    :=P2*FRAC(0.8829867+3.3947688*T);
  M7    :=P2*FRAC(0.3969537+1.1902586*T);
  C5[0]:=1.0;            S5[0]:  =0.0;
  C5[1]:=COS(M5);   S5[1]:  =SIN(M5);  C5[-1]:=C5[1];        S5[-1]:=-S5[1];
FOR I  :=2    TO    5    DO    ADDTHE(C5[I-1],S5[I-1],C5[1],S5[1],C5[I],S5[I]);
   PERTSAT; PERTURA; PERTSUR;
   L:= 360.0*FRAC(0.0388910 + M5/P2
      +    ((5025.2+0.8*T)*T+DL)/1296.0E3    );
   R:= 5.208873 + 0.000041*T    +    DR*1.0E-5;
   B:= ( 227.3 - 0.3*T + DB ) / 3600.0;

  END;   (* JUP200     *)
(*-------------------------------------------------------------------   *)
```

JUP200, как и приведенные в приложении подпрограммы MER200, VEN200, ..., PLU200 определяют для заданного момента T гелиоцентрические координаты — долготу l, широту b и расстояние r — данной планеты относительно эклиптики и точки весеннего равноденствия даты. Время измеряется в юлианских столетиях от эпохи J2000:

$$T = (\mathrm{JD} - 2451545)/36525\,.$$

При вычислении юлианских дат JD нужно учесть, что в машину вводится не всемирное время UT, а эфемеридное ET (или динамическое TDT/TDB). Соответствующая программа вычисления геоцентрических координат Солнца была представлена в гл.2.

Точность вычислений для XIX и XX столетий составляет несколько секунд дуги (табл 5.2). Для более отдаленных времен результаты получаются хуже. Это происходит, во-первых, потому, что используемые теории были созданы в начале века и могли учесть лишь имевшиеся тогда наблюдения. Во-вторых, возмущающие члены, возрастающие как T^3, в программе отбрасываются. Эти члены в настоящее время очень малы ($< 0,1''$), но при больших временных интервалах могут в тысячи раз превосходить указанное значение.

Таблица 5.2. Средние ошибки результатов вычислений по программам MER200—PLU200 в интервале 1750—2250 гг (для Плутона—1890—2100 гг.)

Планета	Δl [$''$]	Δb [$''$]	Δr [10^{-6} а.е]
Меркурий	1,0—1,5	0,7—1,0	1,0— 1,5
Венера	0,5—1,0	0,2—2,5	0,5— 1,5
Солнце-Земля	0,5—2,0	0,0—0,1	0,4— 1,5
Марс	0,5—2,5	0,1—1,0	3 — 10
Юпитер	2 —8	0,7—1,0	20 — 30
Сатурн	2 —11	0,8—1,5	40 — 100
Уран	3 — 8	0,7—1,0	50 — 200
Нептун	3 —40	1,0—2,0	500 —1000
Плутон	1 — 9	0,2—2,2	200 —1000

5.4. Видимые и астрометрические координаты

Хорошие ежегодники, приводя эфемериды планет, обычно указывают, являются *ли приводимые* значения *геометрическими, видимыми* или *астрометрическими* координатами Различиями между этими координатами можно пренебрегать, пока достаточна точность 1′ или хуже. Чтобы получить точность, обеспечиваемую программой, приведенной в предыдущем разделе, нужно знать наиболее важные поправки к координатам планет

В то время как геометрические координаты определяют точку в пространстве, в которой в момент t *находится* планета, видимые и астрометрические координаты дают направление, в котором *наблюдается* планета

Геометрические координаты обычно используются для описания гелиоцентрического положения планеты Дополнительное указание равноденствия однозначно определяет используемую координатную систему Подпрограммы MER200—PLU200 дают *геометрические гелиоцентрические эклиптические координаты для равноденствия даты («на момент счета»).* Исключением является, естественно, подпрограмма SUN200, вычисляющая геоцентрическое положение Солнца.

Геоцентрические положения (исключая расстояние) в большинстве случаев приводятся в форме астрометрических или видимых координат. Причина заключается в том, что вследствие конечной скорости распространения света и собственного движения планеты она наблюдается не на том месте, где располагалась в момент наблюдений. Мы уже отмечали этот эффект при рассмотрении орбит комет. Так как расстояние, в действительности пройденное светом, не является непосредственно наблюдаемой величиной, в геоцентрических эфемеридах также обычно ограничиваются указанием геометрического расстояния.

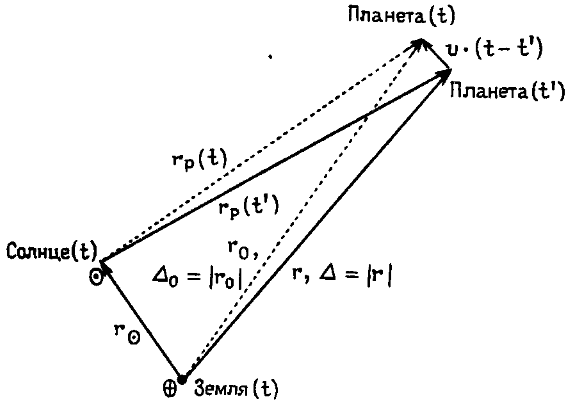


Рис. 5.2. Перемещение планеты за время распространения света.

5.4.1. Аберрация и время распространения света

Астрометрические координаты указывают направление падения наблюдаемого светового луча от планеты на Землю. Этот луч света соединяет точку на Земле в момент наблюдения t с положением планеты в момент испускания света t'. Время распространения света $\tau = t - t'$ для геометрического расстояния

$$\Delta_0 = |\mathbf{r}_0|, \qquad \text{где} \qquad \mathbf{r}_0 = \mathbf{r}_{\odot}(t) + \mathbf{r}_p(t) \tag{5.11}$$

получается равным $\tau \approx \Delta_0/c$ (см. определения отдельных величин в разд. 5.2). Считая, что планета в течение этого времени движется приблизительно прямолинейно, для астрометрических координат в момент наблюдения t получим

$$\mathbf{r} = \mathbf{r}_{\odot}(t) + \mathbf{r}_p(t') \approx \mathbf{r}_{\odot}(t) + \left\{ \mathbf{r}_p(t) - \mathbf{v}_p(t)\frac{\Delta_0}{c} \right\}. \tag{5.12}$$

Разность между *астрометрической* координатой $\mathbf{r}$ и *геометрической* координатой $\mathbf{r}_0$ называется *планетной аберрацией*. Астрометрические координаты можно непосредственно наносить на атлас неба или сравнивать на фотопластинке с известными положениями звезд

Однако, если навести телескоп на параллактической установке на планету с помощью астрометрических координат, то получим отклонение от наблюдаемого положения примерно на 20″ Причины этого отклонения состоят в том, что астрометрические координаты определяют направление падающих на Землю лучей света в системе координат, *неподвижной* относительно Солнца Наблюдатель, напротив, участвует в суточном вращении Земли и ее годичном движении вокруг Солнца Это движение в сочетании с конечной скоростью света приводит к тому, что наблюдатель видит иное направление падения лучей света от планеты Для наглядного объяснения различия между покоящимся и движущимся наблюдателем часто приводят пример, который каждый может проверить на своем опыте капли дождя, падающие на поверхность Земли вертикально, водителю быстро движущегося автомобиля кажутся падающими с той стороны, в которую он движется

Для перехода от неподвижной системы в движущуюся необходимо вносить поправки в координаты не только комет и планет, но и неподвижных звезд В этом случае говорят о *звездной* аберрации В звездных каталогах и атласах неба приводятся положения звезд для наблюдателя в системе, неподвижной относительно Солнца Такие положения соответствуют астрометрическим координатам планет, с которыми их можно сравнивать непосредственно Наблюдатель на Земле, снимая показания разделенного круга, измеряет видимое положение звезды, которое в течение года периодически меняется около своего каталожного значения

Строгий вывод формул звездной аберрации увел бы нас очень далеко, так как речь идет о применении специальной теории относительности Но скорость Земли очень мала по сравнению со скоростью света, поэтому обсуждаемая далее полуклассическая аргументация с достаточной точностью приводит к тому же результату

Если $\mathbf{r}$ — вектор астрометрических координат планеты, то луч света, падающий на Землю, описывается вектором скорости света (рис 5 3)

$$\mathbf{c} = -c \cdot \frac{\mathbf{r}}{|\mathbf{r}|}$$

Соответствующий вектор $\hat{\mathbf{c}}$ в системе, связанной с Землей, получится, если из этого значения вычесть скорость Земли относительно Солнца $\mathbf{v}_\oplus$

$$\hat{\mathbf{c}} = \mathbf{c} - \mathbf{v}_\oplus$$

Наблюдатель при этом видит планету в направлении

$$\hat{\mathbf{r}} = -|\mathbf{r}| \frac{\hat{\mathbf{c}}}{c} = \mathbf{r} + \mathbf{v}_\oplus \cdot \frac{|\mathbf{r}|}{c} \approx$$

$$\approx \mathbf{r} + \mathbf{v}_\oplus \frac{\Delta_0}{c} . \qquad (5.13)$$

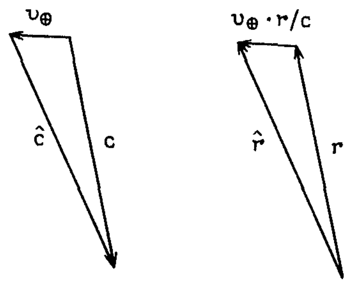


Рис. 5.3. Звездная аберрация

Координаты, найденные таким способом, определяют *видимое* место планеты, если к тому же использованная система координат ориентирована точно в направлении истинного положения земной оси При этом необходимо выбрать равноденствие и учесть нутацию Эти вопросы и будут темой следующего раздела

5 4 2 Скорость планет

Продемонстрируем вычисление вектора гелиоцентрической скорости планеты, используемого для учета различных аберрационных эффектов

Дифференцируя формулы для эклиптических координат (l, b, r) по времени и принимая во внимание только самые большие члены, для производных ($\dot{l}$, $\dot{b}$, $\dot{r}$) получают короткие тригонометрические ряды, зависящие от средней аномалии каждой планеты

$$\dot{l} = \dot{M}\ (1 + 2e \cos (M) + 5/4e^2 \cos (2M) + \ \ ),$$
$$\dot{b} = i\dot{M}\ (-\sin (\omega) \sin (M) + \cos (\omega) \cos (M) + \ \ ), \quad (5\ 14)$$
$$\dot{r} = a\dot{M}(e \sin (M) + e^2/2 \sin (2M) + \ \ )$$

Число членов зависит от эксцентриситета орбиты и ее наклона к эклиптике. Для Юпитера, например, учитывают только следующие члены:

$$\dot{l} = (+14{,}50 + 1{,}41 \cos (M))\ 10^{-4} \text{рад/сут},$$
$$\dot{b} = 0{,}33 \sin (M)\ 10^{-4} \text{ рад/сут},$$
$$\dot{r} = 3{,}66 \sin (M)\ 10^{-4} \text{ а е/сут}$$

По сферическим скоростям ($\dot{l}$, $\dot{b}$, $\dot{r}$) можно теперь вычислить компоненты вектора скорости в эклиптических координатах:

$$
\begin{aligned}
\dot x &= \dot r \cos l \cos b - \dot l\, r \sin l \cos b - \dot b\, r \cos l \sin b, \\
\dot y &= \dot r \sin l \cos b + \dot l\, r \cos l \cos b - \dot b\, r \sin l \sin b, \qquad (5.15)\\
\dot z &= \dot r \sin b + \dot b\, r \cos b.
\end{aligned}
$$

Все этапы вычисления геоцентрических координат с учетом времени распространения света и аберрации собраны в подпрограмме GEOCEN. В зависимости от значения переменной IMODE в качестве поправки к геометрическим координатам употребляются планетная (IMODE=1,2) и звездная (IMODE=2) аберрация. Дополнительно выдается и геоцентрическое геометрическое расстояние Δ_0.

```
(*-----------------------------------------------------------------*)
(*-----------------------------------------------------------------*)
(* GEOCEN: геоцентрические координаты (геометрические  с         *)
(*         учетом времени распространения света)                  *)
(*-----------------------------------------------------------------*)
(* T: время в юлиан. столетиях с J2000; T=(JD-2451545.0)/36525.0   *)
(* LB,BP,RP:  эклиптические гелиоцентрические координаты планеты  *)
(* LS,BS,RS:  эклиптические геоцентрические координаты Солнца     *)
(*                                                                 *)
(* IPLAN:  0=Сол,1=Мер,2=Вен,3=Зем,4=Мар,5=Юпи,6=Сат,7=Ура,        *)
(*                 8=Неп,9=Плу                                     *)
(* IMODE: мода (X,Y,Z) (0=геометрич., 1=астрометрич., 2=видимые)   *)
(* XP,YP,ZP:  эклиптические гелиоцентрические координаты планеты  *)
(* XS,YS,ZS:  эклиптические геоцентрические координаты Солнца     *)
(* X, Y, Z :  эклиптические геоцентрические координаты планеты    *)
(*            геометрические (IMODE=0), астрометрические (IMODE=1)*)
(*            или видимые (IMODE=2)                                *)
(* DELTA0  :  геоцентрическое расстояние (геометрическое)         *)
(*                                                                 *)
(* (все углы в градусах, расстояния в астрономических единицах)   *)
(*-----------------------------------------------------------------*)
PROCEDURE GEOCEN(T, LP,BP,RP, LS,BS,RS: REAL; IPLAN,IMODE: INTEGER;
                 VAR XP,YP,ZP, XS,YS,ZS, X,Y,Z,DELTA0: REAL);

  CONST P2=6.283185307;

  VAR DL,DB,DR, DLS,DBS,DRS, FAC:REAL;
      VX,VY,VZ, VXS,VYS,VZS, M : REAL;

  FUNCTION FRAC(X:REAL):REAL;
    BEGIN  X:=X-TRUNC(X); IF (X<0) THEN X:=X+1; FRAC:=X   END;

  PROCEDURE POSVEL(L,B,R,DL,DB,DR: REAL; VAR X,Y,Z,VX,VY,VZ:REAL);
```

```
  VAR   CL,SL,CB,SB:  REAL;
  BEGIN
    CL:= CS(L);   SL:=SN(L);   CB:=CS(B);   SB:=SN(B);
    X := R*CL*CB;   VX := DR*CL*CB-DL*R*SL*CB-DB*R*CL*SB;
    Y := R*SL*CB;   VY := DR*SL*CB+DL*R*CL*CB-DB*R*SL*SB;
    Z := R*SB;      VZ := DR*SB                 +DB*R*CB;
  END;

BEGIN

  DL:=0.0;  DB:=0.0;  DR:=0.0;  DLS:=0.0;  DBS:=0.0;  DRS:=0.0;

  IF (IMODE>0) THEN

    BEGIN

      M  :=  P2*FRAC(0.9931266+ 99.9973604*T);             (* Солнце    *)
      DLS:= 172.00+5.75*SIN(M);      DRS:= 2.87*COS(M);    DBS:=  0.0;

      (* dl,db в 1e-4 рад./сут, в 1e-4 а.е./сут                        *)
  CASE IPLAN OF
   0:  BEGIN  DL:=0.0; DB:=0.0; DR:=0.0;  END;       (* Солнце       *)
   1:  BEGIN                                          (* Меркурий     *)
   M:=  P2*FRAC(0.4855407+415.2014314*T);
   DL:= 714.00+292.66*COS(M)+71.96*COS(2*M)+18.16*COS(3*M)+
          4.61*COS(4*M)+3.81*SIN(2*M)+2.43*SIN(3*M)+1.08*SIN(4*M);
   DR:=  55.94*SIN(M)+11.36*SIN(2*M)+2.60*SIN(3*M);
   DB:=  73.40*COS(M)+29.82*COS(2*M)+10.22*COS(3*M)+3.28*COS(4*M)
        -40.44*SIN(M)-16.55*SIN(2*M)-5.56*SIN(3*M)-1.72*SIN(4*M);
          END;
     2: BEGIN                                          (* Венера      *)
          M:=  P2*FRAC(0.1400197+162.5494552*T);
          DL:= 280.00+3.79*COS(M);       DR  :=  1.37*SIN(M);
          DB:=    9.54*COS(M)-13.57*SIN(M);
        END;
     3:  BEGIN  DL:=DLS; DR:=DRS; DB:=-DBS;  END;  (* Земля       *)
     4:  BEGIN                                       (* Марс        *)
          M:=  P2*FRAC(0.0538553+53.1662736*T);
          DL:= 91.50+17.07*COS(M)+2.03*COS(2*M);
          DR:= 12.98*SIN(M)+1.21*COS(2*M);
          DB:=   0.83*COS(M)+2.80*SIN(M);
        END;
     5: BEGIN                                        (* Юпитер      *)
          M:=  P2*FRAC(0.0565314+8.4302963*T);
          DL:= 14.50+1.41*COS(M);
          DR:= 3.66*SIN(M);
          DB:= 0.33*SIN(M);
```

```
        END;
      6: BEGIN                                              (* Сатурн   *)
           M :=  P2*FRAC(0.8829867+3.3947688*T);
           DL:=  5.84+0.65*COS(M);
           DR:=  3.09*SIN(M);
           DB:=  0.24*COS(M),
        END;
      7: BEGIN                                              (* Уран     *)
           M :=  P2*FRAC(0.3967117+1.1902849*T);
           DL:=  2.05+0.19*COS(M);
           DR:=  1.86*SIN(M);
           DB:=-0.03*SIN(M);
        END;
      8: BEGIN                                              (* Нептун   *)
           M :=  P2*FRAC(0.7214906+0.6068526*T);
           DL:=  1.04+0.02*COS(M);
           DR:=  0.27*SIN(M);
           DB:=  0.03*SIN(M);
        END;
      9: BEGIN                                              (* Плутон   *)
           M :=  P2*FRAC(0 0385795+0.4026667*T);
           DL:=  0.69+0.34*COS(M)+0.12*COS(2*M)+0.05*COS(3*M);
           DR:=  6.66*SIN(M)+1.64*SIN(2*M);
           DB:=-0.08*COS(M)-0.17*SIN(M)-0.09*SIN(2*M);
        END;
      END;

    END;

  POSVEL  (LS,BS,RS,DLS,DBS,DRS,XS,YS,ZS,VXS,VYS,VZS);
  POSVEL (LP,BP,RP,DL ,DB ,DR, XP,YP,ZP,VX ,VY ,VZ );
  X:=XP+XS;  Y:=YP+YS;   Z:=ZP+ZS;    DELTA0 := SQRT(X*X+Y*Y+Z*Z);
  IF IPLAN=3 THEN BEGIN X:=0.0; Y:=0.0; Z:=0.0; DELTA0:=0.0 END;

  FAC := 0.00578 * DELTA0 * 1E-4;
  CASE IMODE OF
      1: BEGIN X:= X-FAC*VX; Y:=Y-FAC*VY;   Z:=Z-FAC*VZ; END;
      2: BEGIN X:= X-FAC*(VX+VXS);
               Y:= Y-FAC*(VY+VYS);
               Z:= Z-FAC*(VZ+VZS); END;
    END;
  END;
(*-----------------------------------------------------------------*)
```

5.4.3. Нутация

Видимые координаты применяются при наблюдениях на телескопе с разделенным кругом. Так как монтировка телескопа всегда жестко ориентирована вдоль земной оси, используемая система координат должна быть направлена так же. Это означает прежде всего, что видимые координаты всегда даются на равноденствие даты, т е. относительно истинного на данный момент положения экватора, эклиптики и точки весеннего равноденствия. При этом кроме прецессии нужно учитывать еще одно возмущение положения земной оси, о котором мы еще не упоминали при рассмотрении систем координат. В то время как прецессия представляет собой долгопериодическое движение земной оси, под *нутацией* понимают дополнительное малое смещение периодического характера. Из-за нутации *истинный* полюс за 18,6 лет совершает один оборот вокруг *среднего* полюса, движущегося вследствие прецессии. Период нутации определяется временем обращения восходящего узла Ω орбиты Луны. Нутация по долготе ($\Delta\psi$), т. е. разность долгот истинной и средней точки весеннего равноденствия, имеет амплитуду примерно 17″. Одновременно в пределах $\Delta\varepsilon \approx 9''$ меняется и наклонение эклиптики (рис.5.4).

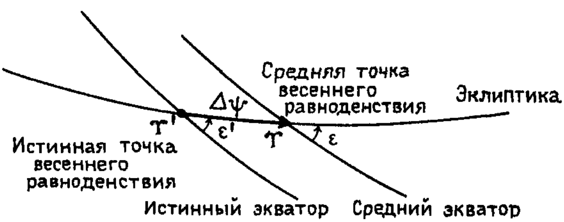


Рис.5.4. Смещение экватора, эклиптики и точки весеннего равноденствия под влиянием нутации.

В общем, выражения этих величин включают примерно сотню членов, из которых для многих задач достаточно нескольких:

$$\Delta\psi = -17{,}200'' \sin(\Omega) + 0{,}206'' \sin(2\Omega) + 0{,}143'' \cos(l') -$$
$$-1{,}319'' \sin(2(F - D + \Omega)) - 0{,}227'' \sin(2(F + \Omega)), \quad (5.16)$$
$$\Delta\varepsilon = +\ 9{,}203'' \cos(\Omega) \quad - 0{,}090'' \cos(2\Omega) +$$
$$+\ 0{,}574'' \cos(2(F - D + \Omega)) + 0{,}098'' \cos(2(F + \Omega)) .$$

Помимо долготы восходящего угла эти члены зависят и от других аргументов (F, D, l'), состоящих из средних долгот и аномалий Солнца и Луны (уравнения (6.2)—(6.5)). Истинные экваториальные координаты (x', y', z') получаются сложением использовавшихся до сих пор средних координат (x, y, z) с небольшой поправкой (Δx, Δy, Δz):

$$\Delta x = -(y \cos \varepsilon + z \sin \varepsilon)\, \Delta\psi,$$

$$\Delta y = x \cos \varepsilon\, \Delta\psi - z\, \Delta\varepsilon, \qquad (5.17)$$

$$\Delta z = x \sin \varepsilon\, \Delta\psi + y\, \Delta\varepsilon.$$

Здесь ε — средний наклон эклиптики из (2.5).

Подпрограмма NUTEQU требует ввода средних экваториальных координат планеты и для момента T преобразует их в истинные координаты. При этом вводимые данные переписываются.

```
(*-----------------------------------------------------------------------*)
(* NUTEQU: Преобразование средних экваториальных координат в            *)
(*         истинные экваториальные координаты                            *)
(*         (члены > 0.1" по МАС 1980)                                    *)
(*         T = (JD-2451545.0)/36525.0                                    *)
(*-----------------------------------------------------------------------*)
PROCEDURE NUTEQU(T:REAL;VAR X,Y,Z:REAL);
  CONST ARC=206264.8062; (* секунд дуги в радиане = 3600*180/pi      *)
        P2=6.283185307;  (* 2*pi                                      *)
  VAR   LS,D,F,N,EPS    : REAL;
        DPSI,DEPS,C,S   : REAL;
        DX,DY,DZ        : REAL;
  FUNCTION FRAC(X:REAL):REAL;
    (* TRUNC для T<-24 заменить на LONG_TRUNC или INT !                *)
    BEGIN  FRAC:=X-TRUNC(X) END;
BEGIN
LS  :=P2*FRAC(0.993133+   99.997306*T);  (* ср. аномалия Солнца      *)
D   :=P2*FRAC(0.827362+1236.853087*T);   (* разн. долгот Луна-Сол.*)
F   :=P2*FRAC(0.259089+1342.227826*T);   (* расст.от узла широты     *)
N   :=P2*FRAC(0.347346-   5.372447*T); (* долгота восх. узла         *)
EPS :=0.4090928-2.2696E-4*T;             (* наклон эклиптики         *)
DPSI:=( -17.200*SIN(N)    - 1.319*SIN(2*(F-D+N))
        - 0.227*SIN(2*(F+N)) + 0.206*SIN(2*N)
        + 0.143*SIN(LS)  ) / ARC;
DEPS:= ( + 9.203*COS(N)   + 0 574*COS(2*(F-D+N))
        + 0.098*COS(2*(F+N))
        - 0.090*COS(2*N)                    ) / ARC;
  C := DPSI*COS(EPS);   S := DPSI*SIN(EPS);
  DX:= -(C*Y+S*Z);      DY:= (C*X-DEPS*Z); DZ := (S*X+DEPS*Y);
  X := X + DX;          Y := Y + DY;       Z  := Z + DZ;
  END;
(*-----------------------------------------------------------------------*)
```

5.5. Программа PLANPOS

Программа PLANPOS позволяет вычислять положения Солнца и девяти планет в заданный момент времени. Из-за ограниченных возможностей подпрограммы PLU200 координаты Плутона могут быть вычислены лишь между 1890 и 2100 гг. Наряду с моментом времени, нужно выбрать и равноденствие, к которому должно быть отнесено вычисленное положение. Программа дает возможность выбрать видимые координаты, а также астрометрические координаты для применяемых в настоящее время равноденствий B1950 или J2000. Видимые координаты вычисляются для истинного равноденствия даты с учетом поправок за прецессию, нутацию, аберрацию и время распространения света. Их можно, например, выставить на разделенном круге телескопа с параллактической установкой или считать с него. Астрометрические координаты используются, если нужно нанести положение планеты на небесные карты соответствующего равноденствия. В них учтены прецессия и время распространения света.

```
(*-----------------------------------------------------------------*)
(*                          PLANPOS                                *)
(*      Гелиоцентрические и геоцентрические координаты планет      *)
(*                      Версия 05.05.89                            *)
(*-----------------------------------------------------------------*)

PROGRAM PLANPOS(INPUT,OUTPUT);

  CONST  J2000 =  0.0;
         B1950 = -0.500002108;

  TYPE REAL33= ARRAY[1..3,1..3] OF REAL;

  VAR DAY, MONTH, YEAR, IPLAN, IMODE, K:  INTEGER;
      HOUR, MODJD, T, TEQX             :  REAL;
      X,Y,Z, XP,YP,ZP, XS,YS,ZS        :  REAL;
      L,B,R, LS,BS,RS, RA,DEC,DELTA,DELTA0: REAL;
      A                                :  REAL33;
      MODE                             :  CHAR;
*)-----------------------------------------------------------------*)
(* Сюда вводятся следующие подпрограммы в указанном порядке:       *)
(* SN, CS, TN, ATN, ATN2, POLAR, GMS                               *)
(* MJD                                                             *)
(* ECLEQU                                                          *)
(* PMATECL, PRECART, NUTEQU                                        *)
(* SUN200,MER200,VEN200,MAR200,JUP200,SAT200,URA200,NEP200,        *)
(* PLU200                                                          *)
(* GEOCEN                                                          *)
(*-----------------------------------------------------------------*)
(* PRINTOUT: Вывод координат одной планеты                         *)
(*-----------------------------------------------------------------*)
PROCEDURE PRINTOUT:INTEGER;L,B,R,RA,DEC,DELTA:REAL);
  VAR H,M: INTEGER;
      S  : REAL;
```

```
  BEGIN
    GMS(L,H,M,S);    WRITE(H:3,M:3,S:5:1);
    GMS(B,H,M,S);    WRITE(H:4,M:3,S:5:1);
    IF IPLAN<4 THEN WRITE(R:11.6)
                ELSE WRITE(R:10:5,'      ');
    GMS(RA,H,M,S);   WRITE(H.4,M:3,S:6:2);
    GMS(DEC,H,M,S);  WRITE(H:4,M:3,S:5:1);
    IF IPLAN<4 THEN WRITE(DELTA:11:6)
                ELSE WRITE(DELTA:10:5,'      ');
  END;
(*----------------------------------------------------------*)

BEGIN (* PLANPOS *)

  WRITELN;
  WRITELN(' PLANPOS геоцентрические и гелиоцентрические координаты
                    планет                                     ');
  WRITELN('         Версия 05.05.89                            ');
  WRITELN('         (c) 1989 Thomas Pfleger,Oliver Montenbruck  ');
  WRITELN;

  REPEAT

    WRITELN;
    WRITELN (' (J) J2000 астрометр.     (B) B1950   астрометр.   ');
    WRITELN (' (S) видимые координаты   (E) конец                ');
    WRITELN;
    WRITE  ('  ввод: '); READLN (MODE);
    WRITELN;

    IF MODE IN ['S','s','J','j','B','b']        THEN

      BEGIN

      (* чтение даты      *)

      WRITE (' дата (день месяц год час) ?       ');
      READLN (DAY,MONTH,YEAR,HOUR); WRITELN; WRITELN; WRITELN;
      WRITELN;
      MODJD:=MJD (DAY,MONTH,YEAR,HOUR);
      T    := (MODJD-51544 5)/36525 0;
      WRITE  ('  дата:  ',  YEAR:5,MONTH:3,DAY.3,HOUR 5 1,'(ET)');
      WRITE       ('JD:':6,(MODJD+2400000.5):12:3,'равноденствие':18);
      CASE MODE OF
       'S','s':   WRITELN('даты');
       'J','j':   WRITELN('J2000');
       'B','b':   WRITELN('B1950');
```

```
END;
 WRITELN;
 (* заглавные строки        *)
 WRITE   ('  ':  10,'l':6,'b' 12,'r':11);
 WRITELN('  ':  7,'ra':5,'dec':13,'delta':13);
 WRITE   ('  ':  9,'  °  ''  ''','   ':3,'   ° ''  ''','':6,     'AE','':4);
   WRITELN (' ':2,'   h   m  s',' ':4,       '  ° ''  ''',' ':6,'AE');

   (* эклиптические координаты Солнца, равноденствие T          *)

   SUN200   (T,LS,BS,RS),

   (* координаты планет                *)

    IF ( (-1.1<T) AND (T<+1 0) )
    THEN K:=9 ELSE K:=8; (* Плутон да/нет       *)

    FOR IPLAN:=0 TO K DO

       BEGIN

    (*  гелиоцентрические эклиптические координаты планет        *)

    CASE IPLAN OF
     1:  MER200(T,L,B,R);   2:  VEN200(T,L,B,R);
     4:  MAR200(T,L,B,R);   5:  JUP200(T,L,B,R);
     6:  SAT200(T,L,B,R);   7:  URA200(T,L,B,R);
     8:  NEP200(T,L,B,R);   9:  PLU200(T,L,B,R);
     0:  BEGIN  L:=0.0,          B:= 0.0;  R:= 0.0;   END;
     3:  BEGIN  L:=LS+180.0;  B:= -BS;  R:= RS;   END;
  END;

(*   геоцентрические эклиптические координаты планет                *)
                        (* с учетом запаздывания                    *)
    IF MODE IN ['S','s'] THEN IMODE:=2 ELSE IMODE:=1;
    GEOCEN (T,  L,B,R,  LS,BS,RS,  IPLAN,IMODE,
                 XP,YP,ZP,   XS,YS,ZS,   X,Y,Z,DELTA0);

    (* прецессия, экваториальные координаты, нутация             *)

     CASE MODE OF
       'J','j':      TEQX:=J2000;  'B','b':      TEQX:=B1950;
      END;

     IF  MODE  IN  ['S','s']
       THEN
        BEGIN ECLEQU(T,X,Y,Z);  NUTEQU(T,X,Y,Z);  END
```

```
          ELSE
           BEGIN
            PMATECL(T,TEQX,A); PRECART (A,XP,YP,ZP);
            PRECART (A,X,Y,Z); ECLEQU (TEQX,X,Y,Z);
          END;

        (* полярные координаты          *)
        POLAR  (XP,YP,ZP,R,B,L);
        POLAR (X,Y,Z,DELTA,DEC,RA);    RA:=RA/15.0;
        (* вывод данных                 *)

        CASE IPLAN OF
          0:  WRITE(' Солнце ');   1:  WRITE(' Меркурий   ');
          2:  WRITE(' Венера ');   3:  WRITE(' Земля      ');
          4:  WRITE(' Марс   ');   5:  WRITE(' Юпитер     ');
          6:  WRITE(' Сатурн ');   7:  WRITE(' Уран       ');
          8:  WRITE(' Нептун ');   9:  WRITE(' Плутон     ');
        END;
      PRINTOUT (IPLAN,L,B,R,RA,DEC,DELTA0); WRITELN;

    END;

    WRITELN;
   WRITELN (' l,b,r  : гелиоцентрич. эклиптич. (геометрич.)     ');
   WRITE   (' ra,dec:  геоцентрич. экваториальн.                 ');
   IF MODE IN ['S','s'] THEN WRITELN('(видимые)')
                         ELSE WRITELN('(астрометрич.)    ');
   WRITELN(' delta   : геоцентрич. расстояние (геометрич.)     ');
       WRITELN;

    END;
  UNTIL MODE IN ['E','e']

END. (* PLANPOS      *)
(*---------------------------------------------------------------
```

PLANPOS вначале запрашивает равноденствие вычисляемых положений. Можно ввести один из трех вариантов: S (видимые координаты, равноденствие даты), J (J2000, астрометрические координаты) и B (B1950, астрометрические координаты). Этот запрос делается после каждого цикла счета. Для окончания работы нужно ввести E.

В следующем примере мы вначале выберем видимые координаты. После соответствующего запроса задается точный момент времени, для которого мы хотим вычислить положение планеты. Нужно учитывать, что время задается в единицах эфемеридного времени (ET). В нашем примере выбрана дата 1 января 1989 0^h ET. Все вводимые данные далее приведены курсивом. В результате PLANPOS выводит желаемые координаты[1]

```
PLANPOS: геоцентрические и гелиоцентрические координаты планет
                       Версия 05.05.89
          (c) 1989 Thomas Pfleger, Oliver Montenbruck

(J) J2000 астрометр.      (B) B1950 астрометр.
(S) видимые координаты   (E) конец
ввод: S
дата (день месяц год час) ?        1 1 1989 0.0

дата: 1989 1 1 0.0(ET) JD: 244527.500 равноденствие даты
```

	l			b			r
	°	'	"	°	'	"	а.е.
Солнце	0	0	0.0	0	0	0.0	0 000000
Меркурий	347	56	35.7	-6	5	21.5	0 370100
Венера	226	5	15.0	1	43	26 9	0.723666
Земля	100	33	9.4	0	0	0.2	0.983309
Марс	60	32	51.9	0	21	19.3	1.49745
Юпитер	64	30	7.8	0	-45	53.2	5.03195
Сатурн	275	6	53.5	0	47	14.5	10 04354
Уран	271	18	42.6	0	-13	47.8	19.31483
Нептун	279	54	35 7	0	55	55.8	30.21917
Плутон	222	55	15.8	15	52	34.5	29.65861
	ra			dec			delta
	h	m	s	°	'	"	а.е.
Солнце	18	45	53.71	-23	1	25.5	0.983309
Меркурий	19	59	16.66	-22	34	11.9	1.175632
Венера	17	7	15.24	-22	3	57.5	1 522289
Земля	0	0	0.00	0	0	0.0	0.000000
Марс	1	13	47.46	8	24	5.3	0.97649
Юпитер	3	38	35.12	18	33	7.5	4.27637
Сатурн	18	24	14.55	-22	36	28.2	11.02274
Уран	18	7	39.26	-23	39	0.8	20.28599
Нептун	18	42	54.14	-22	10	16.6	31.20230
Плутон	15	6	35.07	-1	16	18.2	30.17673

```
l, b, r:  гелиоцентрич.  эклиптич.  (геометрич.)
ra, dec:  геоцентрич.  экваториальн.(видимые)
delta  :  геоцентрич. расстояние (геометрич.)
```

Нужно помнить, что гелиоцентрические эклиптические координаты Солнца, как и геоцентрические экваториальные координаты Земли, равны нулю.

[1] Программа выводит все данные в одной таблице, мы же из-за недостаточной длины строки разделили таблицу на две части. — *Прим. перев.*

Теперь для сравнения вычислим положения планет в тот же момент времени, но для равноденствия J2000. Небольшие различия, получаемые при этом, обусловлены прецессией за 11 лет.

(J) J2000 астрометр. (B) B1950 астрометр.
(S) видимые координаты (E) конец
ввод: *J*
дата (день месяц год час) ? *1 1 1989 0.0*
дата: 1989 1 1 0.0(ET) JD: 244527.500 равноденствие J2000

		l			b		r
	°	′	″	°	′	″	a.e.
Солнце	0	0	0.0	0	0	0.0	0.000000
Меркурий	348	5	49.4	–6	5	22.1	0.370100
Венера	226	14	28.3	1	43	22.9	0.723666
Земля	100	42	22.5	0	0	5.2	0.983309
Марс	60	42	5.0	0	21	24.1	1.49745
Юпитер	64	39	21.0	0	–45	48.3	5.03195
Сатурн	275	16	6.7	0	47	9.5	10.04354
Уран	271	27	55.8	0	–13	52.9	19.31483
Нептун	280	3	48.9	0	55	50.8	30.21917
Плутон	223	4	29.9	15	52	30.6	29.65861
		ra			dec		delta
	h	m	s	°	′	″	a.e.
Солнце	18	46	34.57	–23	0	32.4	0.983309
Меркурий	19	59	56.55	–22	32	12.3	1.175632
Венера	17	7	55.77	–22	4	40.7	1.522289
Земля	0	0	0.00	0	0	0 0	0.000000
Марс	1	14	21.40	8	27	27.8	0 97649
Юпитер	3	39	11.60	18	35	3.6	4.27637
Сатурн	18	24	55.40	–22	35	56.0	11.02274
Уран	18	8	20.47	–23	38	44.9	20.28599
Нептун	18	43	34.77	–22	9	26.4	31.20230
Плутон	15	7	9.55	–1	18	40.7	30.17673

l, b, r: гелиоцентрич. эклиптич. (геометрич.)
ra, dec: геоцентрич. экваториальн. (астрометрич)
delta : геоцентрич. расстояние (геометрич.)

Хотя различия очень малы, ими нельзя пренебрегать. При сравнении с ежегодниками нужно быть внимательными, так как положения планет от *Меркурия до Нептуна могут быть приведены в видимых координатах*, в то время как для Плутона используются астрометрические координаты. Причина в том, что для отождествления этой слабой планеты необходимы небесные атласы или поисковые карты.

(J) J2000 астрометр. (B) B1950 астрометр.
(S) видимые координаты (E) конец
ввод: *E*

При очередном запросе с вводом *E*, программа заканчивает работу.

6. ОРБИТА ЛУНЫ

6.1. Общее описание орбиты Луны

Движение Луны определяется главным образом двумя телами: Солнцем и Землей. Если оценить гравитационные силы, действующие на Луну, то оказывается, что на ее орбиту сильнее влияет не ее близкий сосед Земля, а далекое Солнце. Хотя сила притяжения ослабевает как квадрат расстояния, притяжение Солнца из-за его большой массы превосходит земное. Используя соответствующие значения величин (расстояние Земля—Луна $r \approx 380\ 000$ км, расстояние Солнце—Луна $R \approx 150$ млн км, отношение массы Солнца к массе Земли $M/m \approx 330\ 000$), можно убедиться, что тяготение Солнца примерно в два раза превышает земное:

$$\frac{F_{\odot}}{F_{\oplus}} = \frac{M}{R^2} \Big/ \frac{m}{r^2} \approx 2.$$

Независимо от взаимного расположения Солнца, Земли и Луны, сумма этих сил имеет результирующую компоненту, направленную всегда в сторону Солнца Таким образом, пространственное движение Луны можно описать эллипсом вокруг Солнца с полуосью около 150 млн км, на который накладываются малые месячные колебания. Поскольку основная компонента сил тяготения направлена к Солнцу, орбита, несмотря на колебания, всегда вогнута в сторону центра.

Хотя орбита Луны определяется в основном силой притяжения Солнца, необязательно описывать ее в гелиоцентрических координатах. В конце концов нас интересует движение Луны вокруг Земли. Так как Земля и Луна расположены довольно близко друг к другу, они испытывают почти одинаковые силы тяготения со стороны Солнца. Если бы эти силы были равны, то Солнце вообще не влияло бы на взаимное положение Земли и Луны. Под действием этих сил происходит движение по общей годичной орбите вокруг Солнца, в то время как Земля определяет движение Луны в течение месяца. Разность сил тяготения, действующих на Землю и Луну со стороны Солнца, составляет долю примерно r/R от величины самой силы, что почти в 200 раз меньше силы, действующей на Луну со стороны Земли. Поэтому наиболее целесообразно описывать орбиту Луны в геоцентрической системе координат. Луна при этом движется по месячной кеплеровской орбите вокруг Земли, в той или иной мере возмущаемой Солнцем.

Средний эллипс орбиты имеет эксцентриситет $e = 0{,}055$ и наклонен к эклиптике примерно на $5{,}1°$. Период обращения от перигея до перигея (аномалистический месяц) составляет 27,55 сут. Это почти на двое суток меньше 29,53 сут—времени между двумя последовательными новолуниями или полнолуниями (синодический месяц). Разность связана с видимым перемещением Солнца на фоне неба, которое составляет около 30°в месяц. После завершения оборота вокруг Земли Луне нужно еще двое суток, чтобы наверстать отставание от Солнца и оказаться в той же фазе.

Из-за наличия эксцентриситета орбиты Луна отклоняется от своего среднего положения на $6{,}3^\circ$. Эта наиболее заметная нерегулярность называется *главным неравенством* Она не связана с возмущениями со стороны Солнца, а объясняется только эллиптической формой орбиты. Значение главного неравенства следует из уравнения центра (5.1), приводившегося в связи с разложением в ряд при вычислении орбит планет. Еще Птолемей указывал в Альмагесте, что учет только этого явления не позволяет хорошо описать орбиту Луны. В качестве другой важнейшей поправки он назвал *эвекцию*, возмущение величиной $1{,}3^\circ$, зависящее от взаимного расположения Солнца и Луны. Намного позднее Тихо Браге на основе своих точных наблюдений открыл еще два возмущения (*вариацию* и *годичное неравенство*). Впоследствии открытие Ньютоном закона тяготения стало основой для теоретической интерпретации движения Луны. В 1770 г Майер впервые опубликовал лунные таблицы, которые были настолько точными, что позволяли определять место нахождения и время на море На сегодняшний день известно более тысячи периодических возмущающих членов в описании орбиты Луны.

Наряду с этими более или менее значительными периодическими вариациями движения Луны, влияние Солнца проявляется и в вековых эффектах, важнейшим из которых является вращение плоскости орбиты Луны. При почти постоянном наклоне орбиты за 18,6 лет линия узлов, т. е линия пересечения орбиты Луны и эклиптики, совершает полный оборот на 360°. Это движение *обратное*, и долгота восходящего узла уменьшается относительно точки весеннего равноденствия на $\approx 20^\circ$ в год. Вызвано оно тем, что сила тяготения Солнца постоянно стремится притянуть Луну к эклиптике и установить ось симметрии орбиты перпендикулярно эклиптике. Как у гироскопа, под влиянием силы ось отклоняется, начинает медленное прецессионное движение и за упомянутые 18,6 лет совершает полный оборот вокруг полюса эклиптики.

Положение перигея и пространственная ориентация большой оси орбиты также меняются. Перигей перемещается в течение года в плоскости орбиты (прямым движением) более чем на 40°. Полный оборот точка перигея совершает за 8,85 лет.

Приведем некоторые величины для среднего движения Солнца и Луны, используемые в теории Луны с принятыми в этой теории обозначениями (T = (JD − 2451545)/36525):

- средняя долгота луны (L_0)

$$L_0 = 218{,}31617^\circ + 481267{,}88088^\circ T - 4{,}06'' T^2 =$$
$$= 0{,}60643382^r + 1336{,}85522467^r\, T - 0{,}00000313^r\, T^2, \tag{6.1}$$

- средняя аномалия Луны (l)

$$l = 134{,}96292^\circ + 477198{,}86753^\circ T + 33{,}25'' T^2 =$$
$$= 0{,}37489701^r + 1325{,}55240982^r T + 0{,}00002565^r T^2, \qquad (6.2)$$

* средняя аномалия Солнца (l')

$$l' = 357{,}52543^\circ + 35999{,}04944^\circ T - 0{,}58'' T^2 =$$
$$= 0{,}99312619^r + 99{,}99735956^r T - 0{,}00000044^r T^2, \qquad (6.3)$$

* среднее расстояние Луны от восходящего узла (F)

$$F = 93{,}27283^\circ + 483202{,}01873^\circ T - 11{,}56'' T^2 =$$
$$= 0{,}25909118^r + 1342{,}22782980^r T - 0{,}00000892^r T^2, \qquad (6.4)$$

* и средняя элонгация Луны (D), т. е. разность средних долгот Солнца и Луны

$$D = 297{,}85027^\circ + 445267{,}11135^\circ T + 5{,}15'' T^2 =$$
$$= 0{,}82736186^r + 1236{,}85308708^r T - 0{,}00000397^r T^2. \qquad (6.5)$$

Значения даны в градусной мере и в единицах одного обращения (1^r = 360°). Средняя долгота Луны L_0 отнесена к среднему равноденствию даты, остальные величины являются углами, на которые прецессия не влияет. Долгота восходящего узла (Ω) прямо не используется. Она получается из разности $\Omega = L_0 - F$.

Истинная эклиптическая долгота Луны λ отличается от средней долготы периодическими членами, зависящими от аргументов l, l', F и D·

$$\lambda = L_0 + \Delta\lambda,$$

где

$\Delta\lambda =$	$+22640''$	$\sin(l)$	– (главное неравенство)
	$-4586''$	$\sin(l - 2D)$	+ (эвекция)
	$+2370''$	$\sin(2D)$	+ (вариация)
	$+769''$	$\sin(2l)$	– (главное неравенство)
	$-668''$	$\sin(l)$	– (годичное неравенство)
	$-412''$	$\sin(2F)$	– (приведение к эклиптике)
	$-212''$	$\sin(2l - 2D)$	–
	$-206''$	$\sin(l + l - 2D)$	+
	$+192''$	$\sin(l + 2D)$	–
	$-165''$	$\sin(l' - 2D)$	+
	$+148''$	$\sin(l - l')$	–
	$-125''$	$\sin(D)$	– (параллактическое неравенство)
	$-110''$	$\sin(l + l')$	–
	$-55''$	$\sin(2F - 2D)$	

При невозмущенной орбите эклиптическая широта β, в силу того, что

$$\sin\beta = \sin i \cdot \sin u,$$
$$\beta \approx i \cdot \sin u,$$

зависит от наклона орбиты i и положения на орбите. Здесь u означает аргумент широты, т. е. угол между вектором положения и линией узлов. Широта Луны с учетом возмущений от Солнца в первом приближении выражается несколько иначе:

$$\beta \approx 18520'' \sin(S) + N;$$

здесь $S = F + \Delta S$, причем ΔS состоит из ряда членов, которые в основном совпадают с разложением $\Delta\lambda$. Ограничившись основными различиями, получим

$$\Delta S \approx \Delta\lambda + 412'' \sin(2F) + 541'' \sin(l') .$$

Величина N содержит дополнительные члены изменений широты, вызванные колебаниями наклона орбиты:

$$N = -526'' \sin(F - 2D) + 44'' \sin(l + F - 2D) -$$
$$-31'' \sin(-l + F - 2D) + \ldots . \tag{6.6}$$

Изложенное выше сильно упрощенное описание нужно в первую очередь для общего представления о построении теории движения Луны и вычислении эклиптических долготы и широты Луны. Программа MINI MOON, приведенная в гл.3 без детальных объяснений, использует именно эти уравнения. Вновь рассмотрев эту программу, можно яснее понять отдельные этапы вычислений.

Нужно отметить, что для вычисления эклиптической широты мы сознательно отказались от использования разложения β в ряд. Такой ряд, с важнейшими членами

$$\beta = 18461'' \sin(F) + 1010'' \sin(l + F) + 1000'' \sin(l - F) -$$
$$-624'' \sin(F - 2D) - 199'' \sin(l - F - 2D) -$$
$$-167'' \sin(l + F - 2D) + \ldots ,$$

легче использовать, но вычисления в этом случае потребуют большого времени. Отдельные члены содержат здесь нечетные кратные F, в то время как разложение в ряд $\Delta\lambda$ и ΔS всегда содержит только четные кратные F. Каждому члену в $\Delta\lambda$ соответствует член в разложении ΔS, использующий те же тригонометрические функции. Поэтому при правильной организации для ΔS не нужно дополнительно вычислять тригонометрические члены. Напротив, ряды для $\Delta\lambda$ и $\Delta\beta$ не имеют общих членов, которые можно было бы вычислить вместе.

6.2. Теория Луны Брауна

Разработанная в начале века теория Луны Брауна является, пожалуй, самым известным аналитическим представлением движения Луны. Мы приводим ее в форме, принятой в «Исправленной эфемериде Луны» (Improved Lunar Ephemeris, ILE), изданной Бюро морских ежегодников в 1954 г. Правда, теория содержит намного больше тысячи членов, а в программе MOON используется только часть из них. При этом точность достигает примерно секунды дуги.

Вычисление начинается с определения средних аргументов L, l, l', F и D по формулам (6.1)—(6.5). Время T при этом выражено в юлианских столетиях *эфемеридного времени* начиная с J2000. Средние аргументы испытывают малые долгопериодические изменения, которые как поправки добавляются к вычисленным значениям:

$\Delta L_0('')$	$\Delta l('')$	$\Delta l'('')$	$\Delta F('')$	$\Delta D('')$		
+0,84	+2,94	−6,40	+0,21	+7,24	$\sin(2\pi(0{,}19833+0{,}05611T))$	
+0,31	+0,31	0,0	+0,31	+0,31	$\sin(2\pi(0{,}27869+0{,}04508T))$	
+14,27	+14,27	0,0	+14,27	+14,27	$\sin(2\pi(0{,}16827-0{,}36903T))$	
+7,26	+9,34	0,0	−88,70	+7,26	$\sin(2\pi(0{,}34734-5{,}37261T))$	(6.7)
+0,28	+1,12	0,0	−15,30	+0,28	$\sin(2\pi(0{,}10498-5{,}37899T))$	
+0,24	+0,83	−1,89	+0,24	+2,13	$\sin(2\pi(0{,}42681-0{,}41855T))$	
0,00	0,00	0,00	−1,86	0,00	$\sin(2\pi(0{,}14943-5{,}37511T))$	

Исправленные таким способом средние аргументы орбиты Луны позволяют вычислить пять рядов возмущений:

$\Delta\lambda$ возмущения эклиптической долготы
ΔS, N, $\gamma_1 C$ возмущения эклиптической широты
$\Delta \sin \Pi$ возмущения параллакса.

Все эти ряды имеют общую структуру

$$\left\{ \begin{matrix} \Delta\lambda,\ \Delta S,\ N \\ \gamma_1 C,\ \Delta \sin \Pi \end{matrix} \right\} = \sum_n \left\{ \begin{matrix} a_n,\ b_n,\ c_n \\ d_n,\ e_n \end{matrix} \right\} \left\{ \begin{matrix} \sin \\ \cos \end{matrix} \right\} (p_n l + q_n l' + r_n F + s_n D).$$

Последовательность (p, q, r, s) описывает зависимость отдельных слагаемых от l, l', F и D, т. е. периодичность членов. При вычислениях целесообразно выбирать члены с одинаковыми характеристиками. Приведем небольшую часть общей таблицы возмущающих членов:

$\Delta\lambda$	ΔS	$\gamma_1 C$	$\Delta \sin \Pi$	p	q	r	s
+22639,500″	+22609,07″	+0,079″	+186,5398″	+1	0	0	0
−4586,465″	−4578,13″	−0,077″	+34,3117″	+1	0	0	−2
+2369,912″	+2373,36″	+0,601″	+28,2333″	0	0	0	+2
+769,016″	+767,96″	+0,107″	+10,1657″	+2	0	0	0
−668,146″	−126,98″	−1,302″	−0,3997″	0	+1	0	0
−411,608″	−0,20″	+0,000″	−0,0124″	0	0	+2	0

Из-за различия коэффициентов ряд для величины N не может быть вычислен одновременно с другими рядами. Правда, он состоит всего лишь из нескольких членов.

Большое число возмущающих членов делает оправданным применение некоторых ухищрений при вычислении необходимых угловых функций. Вычисление синусов и косинусов по сравнению с элементарными операциями, т. е. четырьмя арифметическими действиями, требует значительных усилий и времени. Однако совершенно необязательно каждую функцию вычислять отдельно. Используя теорему сложения

$$\cos(\alpha_1 + \alpha_2) = \cos\alpha_1 \cos\alpha_2 - \sin\alpha_1 \sin\alpha_2 ,$$

$$\sin(\alpha_1 + \alpha_2) = \sin\alpha_1 \cos\alpha_2 + \cos\alpha_1 \sin\alpha_2 ,$$

с помощью короткой подпрограммы ADDTHE

```
PROCEDURE ADDTHE(C1,S1,C2,S2:REAL;VAR C,S:REAL);
BEGIN C:=C1*C2-S1*S2; S:=S1*C2+C1*S2; END;
```

можно в первую очередь вычислить синус и косинус от кратных средних аргументов:

$$\mathrm{CO[I,K]} = \begin{cases} \cos(il) & (k=1) \\ \cos(il') & (k=2) \\ \cos(iF) & (k=3) \\ \cos(iD) & (k=4) \end{cases} \qquad \mathrm{SI[I,K]} = \begin{cases} \sin(il) & (k=1) \\ \sin(il') & (k=2) \\ \sin(iF) & (k=3) \\ \sin(iD) & (k=4) \end{cases} .$$

Для l (K = 1) с помощью фрагмента программы

```
CO[0,K]:= 1.0; CO[1,K]:=COS(L);    SI[0,K]:=0.0; SI[1,K]:=SIN(L);
FOR I := 2 TO MAX DO
 ADDTHE(CO[I-1,K],SI[I-1,K],CO[1,K],SI[1;K],CO[I,K],SI[I,K];
FOR I := 1 TO MAX DO
 BEGIN CO[-I,K]:=CO[I,K]; SI[-I,K]:=-SI[I,K]; END;
```

получают все значения $\cos(il)$ и $\sin(il)$ для I = –MAX... + MAX, только два раза вызывая синус и косинус. Процедура TERM, использующая заранее вычисленные значения CO[I,K] и SI[I,K], позволяет довольно просто определить

$$x = \cos(pl + ql' + rF + sD) \text{ и } y = \sin(pl + ql' + rF + sD)$$

для заданных коэффициентов (p, q, r, s).

```
PROCEDURE TERM(P,Q,R,S:INTEGER;VAR X,Y:REAL);
  VAR  I: ARRAY[1..4] OF INTEGER;   K: INTEGER;
  BEGIN
   I[1]  :=P; I[2]:=Q;    I[3]:=R;   I[4]:=S;    X:=1.0;  Y:=0.0;
   FOR K:=1 TO 4 DO
     IF (I[K]<>0) THEN ADDTHE(X,Y,CO[I[K],K],SI[I[K],K],X,Y);
END;
```

На примере вычисления ряда для возмущения широты N (6.6) можно объяснить применение этой процедуры. Процедура SOLARN служит для сложения всех значений названного ряда для переменной N. Каждый отдельный член определяется подпрограммой ADDN, в которую в качестве параметров передаются коэффициенты и характеристические значения (p, q, r, s) *каждого члена.* Затем с помощью TERM вычисляют значения соответствующих угловых функций (N содержит только синусы), умножают их на коэффициенты и прибавляют к N. Естественно, до вызова SOLARN нужно определить все значения массивов CO[I,K] и SI[I,K].

```
 PROCEDURE SOLARN(VAR N:REAL);
 VAR X,Y: REAL;
 PROCEDURE ADDN(COEFFN:  REAL;P,Q,R,S:INTEGER);
  BEGIN TERM(P,Q,R,S,X,Y); N:=N+COEFFN*Y END;
   BEGIN
   N :=0.0;
 ADDN(-526.069,  0,    0, 1,-2);     ADDN(  -3.352, 0,   0, 1,-4);
 ADDN( +44.297, +1,    0, 1,-2);     ADDN(  -6.000,+1,   0, 1,-4);
 ADDN( +20.599, -1,    0, 1, 0);     ADDN( -30.598,-1,   0, 1,-2);
 ADDN( -24.649, -2,    0, 1, 0);     ADDN(  -2.000,-2,   0, 1,-2);
 ADDN( -22.571,  0,   +1, 1,-2);     ADDN( +10.985; 0,  -1, 1,-2);
END;
BEGIN
....здесь вычисляются CO и SI...
 SOLARN(N);    WRITELN(N);
 ...
END;
```

Аналогичным образом работают и процедуры ADDSOL, SOLAR1, SOLAR2 и SOLAR3, вычисляющие ряды $\Delta\lambda$, ΔS, $\gamma_1 C$ и Δ sin Π.

Учет солнечных возмущающих членов, описанный здесь, пока не полон. Это обусловлено тем обстоятельством, что коэффициенты возмущающих членов, строго говоря, немного отличаются от приведенных здесь значений. Каждый коэффициент ряда $\Delta\lambda$, ΔS, N, $\gamma_1 C$ и Δ sin Π является на самом деле функцией нескольких параметров, таких, как, эксцентриситет солнечной орбиты, для которых Браун принял определенные числовые значения. Чтобы вычислить правильные значения этих параметров, каждый член, характеризуемый массивом (p, q, r, s), нужно умножить на

$$(1{,}000002208)^{|p|}(1{,}0-0{,}002495388(T+1))^{|q|}(1{,}000002708+139{,}978\Delta\gamma)^{|r|}$$

Здесь $\Delta\gamma$ суть

$$\begin{aligned}\Delta\gamma = &-0{,}000003332 \sin(2\pi(0{,}59734 - 5{,}37261\,T)) - \\ &-0{,}000000539 \sin(2\pi(0{,}35498 - 5{,}37899\,T)) - \\ &-0{,}000000064 \sin(2\pi(0{,}39943 - 5{,}37511\,T))\end{aligned}$$

(вычисляется в LONG_PERIODIC). Эти поправки очень легко учесть, если при описанном выше вычислении массивов CO[I,K] и SI[I,K] величины

CO[1,1] и SI[1,1] умножить на (1,000002208),
CO[1,2] и SI[1,2] умножить на (1,0 − 0,002495388(T + 1)) и
CO[1,3] и SI[1,3] умножить на (1,000002708 + 139,978$\Delta\gamma$).

При повторном вызове ADDTHE необходимые поправочные множители уже будут учтены в ранее вычисленных значениях тригонометрических функций. Так, SI[3,2] вместо sin ($3l'$) будет содержать значение (1,0 − − 0,002495388 (T + 1))$^{|3|}$ sin ($3l'$). Это небольшое изменение позволяет все члены автоматически умножить на правильный коэффициент.

На этом вычисление различных возмущений движения Луны, вызванных Солнцем, заканчивается. Конечно, Солнце не является единственным телом Солнечной системы, влияющим на Луну. Детальная теория орбиты Луны учитывает и возмущения, вызванные планетами. Поэтому в программе MOON учитывается наиболее существенное влияние Венеры и Юпитера:

$$\begin{aligned}\Delta\lambda_{\text{план}} = &+0{,}82''\sin(0{,}7736 - 62{,}5512T) + 0{,}31''\sin(0{,}0466 - 125{,}1025T) +\\ &+0{,}35''\sin(0{,}5785 - 25{,}1042T) + 0{,}66''\sin(0{,}4591 + 1335{,}8075T) +\\ &+0{,}64''\sin(0{,}3130 - 91{,}5680T) + 1{,}14''\sin(0{,}1480 - 1331{,}2898T) +\\ &+0{,}21''\sin(0{,}5918 + 1056{,}5859T) + 0{,}44''\sin(0{,}5784 - 1322{,}8595T) +\\ &+0{,}24''\sin(0{,}2275 - 5{,}7374T) + 0{,}28''\sin(0{,}2965 + 2{,}6929T) +\\ &+0{,}33''\sin(0{,}3132 + 6{,}3368T).\end{aligned}$$

Эти возмущения эклиптической долготы вычисляются в PLANETARY.

Таким образом, все нужные величины теперь известны. Сложив солнечные и планетарные возмущения средней долготы, получим эклиптическую долготу:

$$\lambda = L_0 + \Delta\lambda + \Delta\lambda_{\text{план}} .$$

L_0 является средней долготой, уже исправленной согласно (6.7). Для получения эклиптической широты нужно сначала сложить ΔS со средним расстоянием узла F:

$$S = F + \Delta S .$$

После этого получим

$$\beta = (1{,}000002708 + 139{,}978\ \Delta\gamma)\ \{18519{,}70'' + \gamma_1 C\} \times$$
$$\times \sin(S) - \{6{,}24''\} \sin(3S) + N.$$

Расстояние до Луны вычисляется по синусу горизонтального **параллакса** Π:

$$r = (1/\sin \Pi)\ R_{\oplus} \quad (\text{радиус Земли } R_{\oplus} \approx 6378{,}14 \text{ км}).$$

sin Π имеет значение

$$\sin \Pi = 0{,}99953253\ (3422{,}7'' + \Delta \sin \Pi)\ \frac{\pi}{180 \cdot 3600''} .$$

Коэффициент

$$\frac{\pi}{180 \cdot 3600''} = \frac{1}{206264{,}81''}$$

служит для перехода от секунд дуги к радианам.

Далее мы приводим всю подпрограмму MOON. Для заданного момента *T* (выраженного в юлианских столетиях эфемеридного времени с J2000) вычисляются геоцентрические эклиптические долгота и широта Луны и расстояние ее от центра Земли, выраженное в единицах радиуса Земли. Координаты вычисляются для среднего равноденствия даты.

```
(*-----------------------------------------------------------------*)
(* MOON: аналитическая теория Луны Брауна (улучшенная эфемерида   *)
(*       Луны), точность около 1"                                  *)
(*                                                                 *)
(*       T:      эфемеридное время в юлианских столетиях с J2000  *)
(*               (T=(JD-2451545.0)/36525 0)                        *)
(*   LAMBDA:геоцентрическая эклипт долгота (равноденст. даты)     *)
(*   BETA  : геоцентрическая эклипт. широта (равноденст. даты)    *)
(*   R     : геоцентрическое расстояние (в ед. земного радиуса)   *)
(*-----------------------------------------------------------------*)

PROCEDURE MOON ( T:REAL; VARLAMBDA,BETA,R: REAL );

  CONST PI2= 6.283185308;     (* 2*pi;      pi = 3.141592654...    *)
        ARC= 206264.81;       (* 3600*180/pi =  сек. дуги в рад.   *)

  VAR DGAM,FAC               : REAL;
      DLAM,N,GAM1C,SINPI:     REAL;
      L0, L, LS, F, D ,S    : REAL;
      DL0,DL,DLS,DF,DD,DS:    REAL;
      CO,SI                  : ARRAY[-6..6,1..4] OF REAL;

  (* дробная часть числа                                           *)
  (* TRUNC для T<-24 заменить на LONG_TRUNC или INT!               *)
  FUNCTION FRAC(X:REAL):REAL;
   BEGIN X:=X-TRUNC(X); IF (X<0) THEN X:=X+1; FRAC:=X END;

  (* вычисление c=cos(a1+a2) и s=sin(a1+a2) по теореме             *)
  (* сложения для c1=cos(a1), s1=sin(a1), c2=cos(a2)                *)
  (* и s2=sin(a2)                                                   *)
  PROCEDURE ADDTHE(C1,S1,C2,S2:REAL;VAR C,S:REAL);
   BEGIN  C:=C1*C2-S1*S2;  S:=S1*C2+C1*S2;  END;

  (* вычисление sin(phi); phi в единицах 1 оборот =360 град.       *)
  FUNCTION SINUS (PHI:REAL):REAL;
   BEGIN  SINUS          :=SIN(PI2*FRAC(PHI));    END;
```

```
(* вычисление долгопериодич.изменений средних элементов      *)
(* l,l',F,D и L0, а также dgamma                              *)
PROCEDURE LONG_PERIODIC ( T: REAL;
       VAR DL0,DL,DLS,DF,DD,DGAM: REAL );
VAR  S1,S2,S3,S4,S5,S6,S7:  REAL;
BEGIN
S1 :=SINUS(0.19833+0.05611*T);  S2:=SINUS(0.27869+0.04508*T);
S3 :=SINUS(0.16827-0.36903*T);  S4:=SINUS(0.34734-5.37261*T);
S5 :=SINUS(0.10498-5.37899*T);  S6:=SINUS(0.42681-0.41855*T);
S7 :=SINUS(0.14943-5.37511*T);
DL0:= 0.84*S1+0.31*S2+14.27*S3+7.26*S4+0.28*S5+0.24*S6;
DL := 2.94*S1+0.31*S2+14.27*S3+9.34*S4+1.12*S5+0.83*S6;
DLS:=-6.40*S1                                    -1.89*S6;
DF := 0.21*S1+0.31*S2+14.27*S3-88.70*S4-15.30*S5+0.24*S6
                                                  -1.86*S7;
DD := DL0-DLS;
  DGAM:=-3332E-9  *SINUS(0.59734-5.37261*T)
          -539E-9 *SINUS(0.35498-5.37899*T)
           -64E-9 *SINUS(0.39943-5.37511*T);
 END;

(* INIT: вычисление средних элементов и их sin и cos          *)
(*   l аномалия Луны          l' аномалия Солнца               *)
(*   F расст. Луны от узла  D  элонгация Луны                  *)

PROCEDURE INIT;
 VAR I,J,MAX    :  INTEGER;
     T2,ARG,FAC:  REAL;
BEGIN
T2    :=T*T;
DLAM:=0;  DS:=0;  GAM1C:=0;  SINPI:=3422.7000;
LONG_PERIODIC ( T, DL0,DL,DLS,DF,DD,DGAM );
L0  :=  PI2*FRAC(0.60643382+1336.85522467*T-0.00000313*T2)
         + DL0/ARC;
L   :=PI2*FRAC(0.37489701+1325.55240982*T+0.00002565*T2)
         + DL /ARC;
LS  :=  PI2*FRAC(0.99312619+  99.99735956*T-0.00000044*T2)
         + DLS/ARC;
F   :=  PI2*FRAC(0.25909118+1342.22782980*T-0.00000892*T2)
         + DF /ARC;
D   :=  PI2*FRAC(0.82736186+1236.85308708*T-0.00000397*T2)
         + DD /ARC;
FOR I:= 1 TO 4 DO
 BEGIN
  CASE I OF
   1:  BEGIN ARG:=L;  MAX:=4;  FAC:=1.000002208;                END;
   2:  BEGIN ARG:=LS; MAX:=3;  FAC:=0.997504612-0.002495388*T;  END;
```

```
  3:  BEGIN  ARG:=F;  MAX:=4;  FAC:=1.000002708+139.978*DGAM;     END;
  4:  BEGIN  ARG:=D;  MAX:=6;  FAC:=1.0;                          END;
END;
 CO[0,I]:=1.0;     CO[1,I]:=COS(ARG)*FAC;
 SI[0,I]:=0.0;     SI[1,I]:=SIN(ARG)*FAC;
 FOR J := 2 TO MAX DO
 ADDTHE(CO[J-1,I],SI[J-1,I],CO[1,I],SI[1,I],CO[J,I],SI[J,I]);
          FOR  J := 1 TO MAX DO
             BEGIN    CO[-J,I]:=CO[J,I];     SI[-J,I]:=-SI[J,I];     END;
         END;
 END;

(* TERM вычисление X=cos(p*arg1+q*arg2+r*arg3+s*arg4) и              *)
(* Y=sin(p*arg1+q*arg2+r*arg3+s*arg4)                                *)
PROCEDURE TERM(P,Q,R,S:INTEGER;VAR X,Y:REAL);
 VAR I:  ARRAY[1..4] OF INTEGER;   K: INTEGER;
 BEGIN
  I[1]  :=  P;   I[2]:=Q;   I[3].=R;   I[4].=S;      X:=1.0;   Y:=0.0;
  FOR K:= 1 TO 4 DO
    IF  (I[K]<>0)  THEN      ADDTHE(X,Y,CO[I[K],K],SI[I[K],K],X,Y);
 END;

PROCEDURE  ADDSOL(COEFFL,COEFFS,COEFFG,COEFFP:REAL;
                 P,Q,R,S:INTEGER);
 VAR X,Y: REAL;
 BEGIN
  TERM(P,Q,R,S,X,Y);
  DLAM:=  DLAM +COEFFL*Y;  DS :=DS    +COEFFS*Y;
  GAM1C:=GAM1C+COEFFG*X; SINPI:=SINPI+COEFFP*X;
END;

PROCEDURE SOLAR1;
 BEGIN
    ADDSOL(   13.902,      14.06,    -0.001,     0.2607,  0,  0,  0,   4);
    ADDSOL(    0.403,      -4.01,    +0.394,     0.0023,  0,  0,  0,   3);
    ADDSOL( 2369.912,    2373.36,    +0.601,    28 2333,  0,  0,  0,   2);
    ADDSOL( -125.154,    -112.79,    -0.725,    -0.9781,  0,  0,  0,   1);
    ADDSOL(    1 979,       6.98,    -0.445,     0.0433,  1,  0,  0,   4);
    ADDSOL(  191.953,     192 72,    +0.029,     3.0861,  1,  0,  0,   2);
    ADDSOL(   -8.466,     -13.51,    +0.455,    -0.1093,  1,  0,  0,   1);
    ADDSOL(22639.500,   22609.07,    +0.079,   186.5398,  1,  0,  0,   0);
    ADDSOL(   18.609,       3.59,    -0.094,     0.0118,  1,  0,  0,  -1);
    ADDSOL(-4586.465,   -4578.13,    -0.077,    34.3117,  1,  0,  0,  -2);
    ADDSOL(   +3.215,       5.44,    +0.192,    -0.0386,  1,  0,  0,  -3);
    ADDSOL(  -38.428,     -38.64,    +0.001,     0.6008,  1,  0,  0,  -4);
    ADDSOL(   -0.393,      -1.43,    -0.092,     0.0086,  1,  0,  0,  -6);
```

```
    ADDSOL(   -0.289,    -1.59,   +0.123,   -0.0053,  0,  1,  0,   4);
    ADDSOL(  -24.420,   -25.10,   +0.040,   -0.3000,  0,  1,  0,   2);
    ADDSOL(   18.023,    17.93,   +0.007,    0.1494,  0,  1,  0,   1);
    ADDSOL(-668.146,  -126.98,   -1.302,   -0.3997,  0,  1,  0,   0);
    ADDSOL(    0.560,     0.32,   -0.001,   -0.0037,  0,  1,  0,  -1);
    ADDSOL(-165.145,  -165.06,   +0.054,    1.9178,  0,  1,  0,  -2);
    ADDSOL(   -1.877,    -6.46,   -0.416,    0.0339,  0,  1,  0,  -4);
    ADDSOL(    0.213,     1.02,   -0.074,    0.0054,  2,  0,  0,   4);
    ADDSOL(   14.387,    14.78,   -0.017,    0.2833,  2,  0,  0,   2);
    ADDSOL(   -0.586,    -1.20,   +0.054,   -0.0100,  2,  0,  0,   1);
    ADDSOL(  769.016,   767.96,   +0.107,   10.1657,  2,  0,  0,   0);
    ADDSOL(   +1.750,     2.01,   -0.018,    0.0155,  2,  0,  0,  -1);
    ADDSOL(-211.656,  -152.53,   +5 679,   -0.3039,  2,  0,  0,  -2);
    ADDSOL(   +1.225,     0 91,   -0 030,   -0.0088,  2,  0,  0,  -3);
    ADDSOL(  -30.773,   -34.07,   -0 308,    0.3722,  2,  0,  0,  -4);
    ADDSOL(   -0.570,    -1.40,   -0 074,    0.0109,  2,  0,  0,  -6);
    ADDSOL(   -2.921,   -11.75,   +0.787,   -0.0484,  1,  1,  0,   2);
    ADDSOL(   +1.267,     1.52,   -0 022,    0.0164,  1,  1,  0,   1);
    ADDSOL(-109.673,  -115.18,   +0.461,   -0.9490,  1,  1,  0,   0);
    ADDSOL(-205.962,  -182.36,   +2 056,   +1.4437,  1,  1,  0,  -2);
    ADDSOL(    0.233,     0 36,    0.012,   -0.0025,  1,  1,  0,  -3);
    ADDSOL(   -4.391,    -9 66,   -0.471,    0.0673,  1,  1,  0,  -4);
  END;

PROCEDURE SOLAR2;
  BEGIN
    ADDSOL(   0.283,     1.53,   -0.111,   +0.0060,  1,  -1,  0,  +4);
    ADDSOL(  14.577,    31.70,   -1.540,   +0.2302,  1,  -1,  0,   2);
    ADDSOL( 147.687,   138.76,   +0 679,   +1.1528,  1,  -1,  0,   0);
    ADDSOL(  -1.089,     0.55,   +0 021,    0.0   ,  1,  -1,  0,  -1);
    ADDSOL(  28.475,    23.59,   -0.443,   -0.2257,  1,  -1,  0,  -2);
    ADDSOL(  -0.276,    -0.38,   -0.006,   -0.0036,  1,  -1,  0,  -3);
    ADDSOL(   0.636,     2.27,   +0.146,   -0.0102,  1,  -1,  0,  -4);
    ADDSOL(  -0.189,    -1.68,   +0.131,   -0.0028,  0,   2,  0,   2);
    ADDSOL(  -7.486,    -0.66,   -0.037,   -0.0086,  0,   2,  0,   0);
    ADDSOL(  -8.096,   -16.35,   -0.740,    0.0918,  0,   2,  0,  -2);
    ADDSOL(  -5.741,    -0 04,    0.0  ,   -0.0009,  0,   0,  2,   2);
    ADDSOL(   0.255,     0.0 ,    0.0  ,    0.0   ,  0,   0,  2,   1);
    ADDSOL(-411.608,    -0.20,    0.0  ,   -0.0124,  0,   0,  2,   0);
    ADDSOL(   0.584,     0.84,    0.0  ,   +0.0071,  0,   0,  2,  -1);
    ADDSOL( -55.173,   -52.14,    0.0  ,   -0.1052,  0,   0,  2,  -2);
    ADDSOL(   0.254,     0.25,    0.0  ,   -0.0017,  0,   0,  2,  -3);
    ADDSOL(  +0.025,    -1.67,    0.0  ,   +0.0031,  0,   0,  2,  -4);
    ADDSOL(   1.060,     2.96,   -0.166,    0.0243,  3,   0,  0,  +2);
    ADDSOL(  36.124,    50.64,   -1.300,    0.6215,  3,   0,  0,   0);
    ADDSOL( -13 193,   -16.40,   +0.258,   -0.1187,  3,   0,  0,  -2);
```

```
      ADDSOL( -1.187,     -0.74,  +0.042,   0.0074,  3,  0 , 0, -4);
      ADDSOL( -0.293,     -0.31,  -0.002,   0.0046,  3,  0 , 0, -6);
      ADDSOL( -0.290,     -1.45,  +0.116,  -0.0051,  2,  1 , 0,  2);
      ADDSOL( -7.649,    -10.56,  +0.259,  -0.1038,  2,  1 , 0,  0);
      ADDSOL( -8.627,     -7.59,  +0.078,  -0.0192,  2,  1 , 0, -2);
      ADDSOL( -2.740,     -2.54,  +0.022,   0.0324,  2,  1 , 0, -4);
      ADDSOL(  1.181,      3.32,  -0.212,   0 0213,  2, -1 , 0, +2);
      ADDSOL(  9.703,     11.67,  -0.151,   0.1268,  2  -1 , 0,  0);
      ADDSOL( -0.352,     -0.37,  +0.001,  -0.0028,  2, -1 , 0, -1);
      ADDSOL( -2.494,     -1.17,  -0.003,  -0.0017,  2, -1 , 0, -2);
      ADDSOL(  0.360,      0.20,  -0.012,  -0.0043,  2, -1 , 0, -4);
      ADDSOL( -1.167,     -1.25,  +0.008,  -0.0106,  1,  2 , 0,  0);
      ADDSOL( -7.412,     -6.12,  +0.117,   0.0484,  1,  2 , 0, -2);
      ADDSOL( -0.311,     -0.65,  -0.032,   0.0044,  1,  2 , 0, -4);
      ADDSOL( +0.757,      1 82,  -0.105,   0.0112,  1, -2 , 0,  2);
      ADDSOL( +2.580,      2 32,  +0 027,   0.0196   1, -2 , 0,  0);
      ADDSOL( +2.533,      2.40,  -0.014,  -0.0212,  1, -2 , 0, -2);
      ADDSOL( -0.344,     -0.57,  -0.025,  +0 0036,  0,  3 , 0, -2);
      ADDSOL( -0.992,     -0.02,   0.0   ,  0.0   ,  1,  0 , 2,  2);
      ADDSOL(-45.099,     -0.02,   0.0   , -0.0010,  1,  0 , 2,  0);
      ADDSOL( -0.179,     -9 52,   0.0   , -0.0833,  1,  0 , 2, -2);
      ADDSOL( -0.301,     -0 33,   0.0   ,  0.0014,  1,  0,  2, -4);
      ADDSOL( -6.382,     -3.37,   0.0   , -0.0481,  1,  0, -2,  2);
      ADDSOL( 39.528,     85.13,   0.0   , -0.7136,  1,  0, -2,  0);
      ADDSOL(  9.366,      0.71,   0.0   , -0.0112,  1,  0, -2, -2);
      ADDSOL(  0.202,      0.02,   0.0   ,  0.0   ,  1,  0, -2, -4);
   END;

 PROCEDURE SOLAR3;
   BEGIN
      ADDSOL(  0.415,      0.10,   0.0   ,  0.0013,  0,  1,  2,  0);
      ADDSOL( -2.152,     -2.26,   0.0   , -0.0066,  0,  1,  2, -2);
      ADDSOL( -1.440,     -1.30,   0.0   , +0.0014,  0,  1, -2,  2);
      ADDSOL(  0.384,     -0.04,   0.0   ,  0.0   ,  0,  1, -2, -2);
      ADDSOL( +1.938,     +3.60,  -0.145,  +0.0401,  4,  0,  0,  0);
      ADDSOL( -0.952,     -1.58,  +0.052,  -0.0130,  4,  0,  0, -2);
      ADDSOL( -0.551,     -0.94,  +0.032,  -0.0097,  3,  1,  0,  0);
      ADDSOL( -0.482,     -0.57,  +0.005,  -0.0045,  3,  1,  0, -2);
      ADDSOL(  0.681,      0.96,  -0.026,   0.0115,  3, -1,  0,  0);
      ADDSOL( -0.297,     -0.27,   0.002,  -0.0009,  2,  2,  0, -2);
      ADDSOL(  0.254,     +0.21,  -0.003,   0.0   ,  2, -2,  0, -2);
      ADDSOL( -0.250,     +0.22,   0.004,   0.0014,  1,  3,  0, -2);
      ADDSOL( -3.996,      0.0 ,   0.0   , +0.0004,  2,  0,  2,  0);
      ADDSOL(  0.557,     -0.75,   0.0   , -0.0090,  2,  0,  2, -2);
      ADDSOL( -0.459,     -0.38,   0.0   , -0.0053,  2,  0, -2,  2);
      ADDSOL( -1.298,      0.74,   0.0   , +0.0004,  2,  0, -2,  0);
      ADDSOL(  0.538,      1.14,   0.0   , -0.0141,  2,  0, -2, -2);
```

```
    ADDSOL(  0.263,     0.02,    0.0   ,   0.0    , 1,  1,  2,  0);
    ADDSOL(  0.426,    +0.07,    0.0   ,  -0.0006, 1,  1, -2, -2);
    ADDSOL( -0.304,    +0.03,    0.0   ,  +0.0003, 1, -1,  2,  0);
    ADDSOL( -0.372,    -0.19,    0.0   ,  -0.0027, 1, -1, -2,  2);
    ADDSOL( +0.418,     0.0 ,    0.0   ,   0.0    , 0,  0,  4,  0);
    ADDSOL( -0.330,    -0.04,    0.0   ,   0.0    , 3,  0,  2,  0);
  END;

(* часть N возмущений эклиптической широты                          *)
PROCEDURE SOLARN(VAR N: REAL);
  VAR X,Y: REAL;
  PROCEDURE ADDN(COEFFN:REAL;P,Q,R,S:INTEGER);
    BEGIN TERM(P,Q,R,S,X,Y); N:=N+COEFFN*Y END;
  BEGIN
    N :=  0.0;
    ADDN(-526.069,   0,  0,1,-2); ADDN(   -3.352,  0,  0,1, -4);
    ADDN( +44.297,  +1,  0,1,-2); ADDN(   -6.000, +1,  0,1, -4);
    ADDN( +20.599,  -1,  0,1, 0); ADDN(  -30.598, -1,  0,1, -2);
    ADDN( -24.649,  -2,  0,1, 0); ADDN(   -2.000, -2,  0,1, -2);
    ADDN( -22.571,   0, +1,1,-2); ADDN(  +10.985,  0, -1,1, -2);
  END;

(* возмущения эклиптической долготы Венерой и Юпитером               *)
PROCEDURE PLANETARY(VAR DLAM:REAL);
  BEGIN
   DLAM:= DLAM
  +0.82*SINUS(0.7736   -  62.5512*T)  +0.31*SINUS(0.0466    -125.1025*T)
  +0 35*SINUS(0.5785   -  25.1042*T)  +0.66*SINUS(0.4591+1335.8075*T)
  +0.64*SINUS(0.3130   -  91.5680*T)  +1.14*SINUS(0.1480+1331.2898*T)
  +0.21*SINUS(0.5918  +1056.5859*T)+0.44*SINUS(0.5784+1322.8595*T)
  +0.24*SINUS(0.2275      -5.7374*T)  +0.28*SINUS(0.2965+2.6929*T)
  +0.33*SINUS(0.3132      +6 3368*T);
END;

BEGIN
  INIT; SOLAR1, SOLAR2; SOLAR3; SOLARN(N);  PLANETARY(DLAM);
  LAMBDA:= 360.0*FRAC( (L0+DLAM/ARC) / PI2 );
  S    := F + DS/ARC;
  FAC .= 1.000002708+139.978*DGAM;
  BETA:= ( FAC*(18518.511+1.189+GAM1C)*SIN(S)
          -6.24*SIN(3*S)+N  )  /  3600.0;
  SINPI:= SINPI  *  0.999953253;
  R     := ARC / SINPI;
END;
(*-------------------------------------------------------------------*)
```

Процедура MOON дополняется подпрограммой MOONEQU, которая вместо средних эклиптических координат позволяет вычислить истинные экваториальные координаты — прямое восхождение и склонение.

```
(*-----------------------------------------------------------------*)
(* MOONEQU:экваториальные координаты Луны                          *)
(*          (прямое восхождение RA и склонение DEC в  град.,       *)
(*          R в единицах радиуса Земли)                            *)
(*          T в юлианских столетиях с J2000                        *)
(*          (T:= (JD - 2451545.0)/36525 )                          *)
(*          Координаты отнесены к равноденствию даты               *)
(*-----------------------------------------------------------------*)
PROCEDURE MOONEQU(T:REAL;VAR RA,DEC,R:REAL);
 VAR L,B,X,Y,Z: REAL;
 BEGIN
   MOON(T,L,B,R);       (* эклипт. координаты Луны                 *)
   CART(R,B,L,X,Y,Z);   (* (ср. равноденствие даты)                *)
   ECLEQU(T,X,Y,Z);     (* преобраз.в экваториальн.координаты      *)
   NUTEQU(T,X,Y,Z);     (* нутация                                 *)
  POLAR(X,Y,Z,R,DEC,RA);
 END;
(*-----------------------------------------------------------------*)
```

6.3. Приближение полиномами Чебышева

Несмотря на все искусственные приемы вычисления рядов возмущений, расчет координат Луны с помощью подпрограммы MOON требует довольно больших усилий. Это особенно неудобно в том случае, когда надо вычислить сразу много положений Луны. Например, в следующей главе мы будем рассматривать покрытия звезд Луной, для чего методом итераций нужно найти время соединения Луны с определенной звездой. Для успешного решения такой задачи, требующей большого времени счета, нужны еще две подпрограммы, в которых дается приближение (аппроксимация) координат небесного тела в форме, удобной для практического использования. При этом время вычисления всегда экономится, если вычисляется больше координат, чем используется для построения приближения.

С простым примером аппроксимации мы уже познакомились в гл.3, представляя ход изменений высоты Солнца и Луны в виде набора парабол. В этом случае по трем опорным точкам определялась парабола второго порядка, представляющая ход функции внутри заданного интервала. Вообще, для n различных точек (x_i, y_i), где $i = 1...n$, можно единственным образом определить полином $P_n(x)$ порядка $n - 1$, которому соответствует кривая, в точности проходящая через заданные точки. Для нахождения такого полинома, существует множество методов. Назовем лишь методы Ньютона и Лагранжа.

$$x_k^{n+1} = \frac{b-a}{2}\cos\left[\pi\frac{2k-1}{2n+2}\right] + \frac{(a+b)}{2},$$

$$T_j(\hat{x}_k^{n+1}) = \cos\left[j\pi\frac{2k-1}{2n+2}\right].$$

Понятно, что для вычисления коэффициентов c_j функцию f нужно вычислить в $(n+1)$ точках. Они жестко задаются как нулевые точки T_{n+1}, а не произвольно выбираются, как в интерполяции Лагранжа. В этом и кроется секрет эффективности приближения Чебышева. Выражение f^* *является не только полиномом n-го порядка, который совпадает с* f в $(n+1)$ точках ($f^*(x_k^{n+1}) = f(x_k^{n+1})$). Выбор опорных точек, которые на краях интервалов расположены теснее, чем в середине, гарантирует всюду равномерную ошибку аппроксимации. Это особенно наглядно видно, если при вычислении f^* суммировать не все члены (6.12). Если отбросить, скажем, член самого высокого порядка c_nT_n, можно не сомневаться, что результирующая ошибка не превысит $|c_n|$, поскольку $|T_n| \le 1$ во всем интервале $[a, b]$. Такая оценка невозможна в других методах интерполяции.

Конечно, аппроксимация Чебышева может быть использована в такой форме, если мы можем вычислить необходимые значения функции во всех заданных точках. Но мы хотим разложить в ряд координаты небесного тела, которые в принципе известны в любой момент времени. Поэтому это ограничение для нас не играет никакой роли.

Вычисления по формуле (6.13) в предлагаемой форме все равно требуют длительного счета. Для каждого коэффициента разложения c_j нужно вычислить сумму $(n+1)$ косинусов. Поэтому для вычисления аппроксимации n-го порядка возникает необходимость вычислить косинус $(n+1)^2$ раз. Мы попытались найти в этом случае лучшее решение.

Например, точки $\hat{x}_k^{n+1}$ можно вычислить рекурсивным методом. Для $k = 1,2,\ldots,(n+1)$ получим

$$\hat{x}_k^{n+1} = \cos\left[\frac{\pi}{2n+2}\right], \cos\left[\frac{3\pi}{2n+2}\right], \cos\left[\frac{5\pi}{2n+2}\right], \ldots,$$

$$\cos\left[\frac{(2n+1)\pi}{2n+2}\right].$$

Последовательность этих значений можно получить очень быстро, предполагая в соотношении (6.9) $\varphi = \pi/(2n+2)$. Как программа это может быть записано следующим образом:

```
PHI  := PI/(2*N+2);                    (*  h(k)=cos(pi*k/N/2)
H[0]  := 1;  H[1]:=COS(PHI);
FOR K:= 2 TO (2*N+1) DO H[K]:=2*H[1]*H[K-1]-H[K-2];
```

Элемент массива H[2*K-1] содержит нулевую точку $\hat{x}_k^{n+1}$. Совершенно аналогично вычисляются значения

$$\cos(j\pi(2k - 1)/(2n + 2)).$$

Здесь и далее мы будем исходить из разложений в ряд положений Луны в интервале времени между T_a и T_b. Опорные точки T[K], а также долгота L[K], широта B[K] и расстояние R[K] в эти моменты определяются следующим образом:

```
BMA := (TB-TA)/2.0;                          (*полуширина интервала    *)
BPA := (TA+TB)/2.0;                          (* середина интервала     *)
FOR K:= 1 TO N+1 DO T[K] :=H[2*K-1]*BMA+BPA    (* опорные точки *)
FOR K:= 1 TO N+1 DO MOON(T[K],L[K],B[K],R[K]);  (*значения      *)
                                                (* функции      *)
FOR K:= 2 TO N+1 DO
  IF (L[K-1]<L[K]) THEN L[K]:=L[K] -360.0;
```

Последняя команда обеспечивает устранение разрывности долготы при скачке от 360 к 0°. Чтобы результат разложения не получился ошибочным, от скачка обязательно нужно избавиться.

Чтобы каждый раз не задавать границы интервала, порядок и вектор коэффициентов полинома, в основной программе определяют структуру данных в виде

```
CONST MAX_TP_DEG= 13;
TYPE TPOLYNOM  = RECORD                     (*  полином Чебышева   *)
                 M  : INTEGER;              (*  степень            *)
                 A,B: REAL;                 (*  интервал           *)
                 C  : ARRAY [0..MAX_TP_DEG] OF REAL (* коэффиц.    *)
             END;
```

Процедура T_FIT_MOON вычисляет ряд Чебышева для координат Луны. Вызываемая при этом программа T_FIT_LBR, вообще говоря, может быть использована для разложения в ряд координат Солнца и планет. В этом случае вместо параметров POSITION вводят программу SUN200 или одну из подпрограмм MER200,...,PLU200 из гл.5.

```
(*-----------------------------------------------------------------*)
(* T_FIT_LBR: Разложение в ряд Чебышева координат Луны или         *)
(*        планеты (разложение в ряд долготы, широты и расстояния   *)
(*                                                                 *)
(*          POSITION: процедура вычисления координат L,B,R         *)
(*          TA      : начало интервала разложения                  *)
(*          TB      : конец интервала разложения                   *)
(*          N       : порядок полинома; (N<=MAX_TP_DEG)            *)
(*          L_POLY,B_POLY,R_POLY: полином Чебышева для L,B,R       *)
(*                                                                 *)
(* Примечания:                                                     *)
(*     интервал [TA,TB] должен быть короче одного обращения!       *)
```

```
(*    разложение в ряд можно использовать только для                *)
(*    гелиоцентрических координат планет или геоцентрических        *)
(*    координат Луны!                                               *)
(*----------------------------------------------------------------- *)
PROCEDURE T_FIT_LBR ( PROCEDURE POSITION (T:REAL;
                          VAR LL,BB,RR: REAL);
                          TA,TB: REAL; N: INTEGER;
                          VAR L_POLY,B_POLY,R_POLY: TPOLYNOM);
  CONST PI= 3.1415926535898;
        NDIM= 27;
  VAR   I,J,K           : INTEGER;
        FAC,BPA,BMA,PHI: REAL;
        T,H,L,B,R       : ARRAY[0..NDIM] OF REAL;
  BEGIN
   IF (NDIM<2*MAX_TP_DEG+1)
        THEN WRITELN(' NDIM слишком мало в T_FIT_LBR');
   IF (N>MAX_TP_DEG) THEN WRITELN(' N слишком велико в T_FIT_LBR');
   L_POLY.M:= N;  B_POLY.M:= N;  R_POLY.M:= N;
   L_POLY.A := TA;  B_POLY.A:= TA;  R_POLY.A:= TA;
   L_POLY.B := TB;  B_POLY.B:= TB;  R_POLY B:= TB;
   BMA      := (TB-TA)/2.0;              BPA:= (TB+TA)/2.0;
   FAC      := 2.0/(N+1);
   PHI  := PI/(2*N+2);                       (* h(k)=cos(pi*k/N/2)   *)
   H[0] := 1.0;    H[1]:=COS(PHI);
   FOR I:= 2  TO  (2*N+1)  DO  H[I]:=2*H[1]*H[I-1]-H[I-2];
   FOR K:= 1 TO N+1 DO T[K] := H[2*K-1]*BMA+BPA; (* опорн. точки    *)
   FOR K:= 1  TO  N+1  DO  POSITION(T[K],L[K],B[K],R[K]);
   FOR K:= 2 TO N+1 DO                   (* L становится непрерыв.  *)
    IF (L[K-1]<L[K]) THEN L[K]:=L[K]-360.0; (* на [-360,+360] !     *)
    FOR J:= 0 TO N DO ( * вычисл. коэфф. Чебышева C(j)              *)
     BEGIN
        PHI  := PI*J/(2*N+2);       H[1]   := COS(PHI);
        FOR I:= 2 TO (2*N+1) DO H[I]   := 2*H[1]*H[I-1]-H[I-2];
        L_POLY.C[J]:=0.0; B_POLY.C[J]:=   0.0;    R_POLY.C[J]:=0.0;
        FOR K:= 1 TO N+1 DO
         BEGIN
      L_POLY.C[J]:=  L_POLY.C[J]   +   H[2*K-1]*L[K];
      B_POLY.C[J]:=  B_POLY.C[J]   +   H[2*K-1]*B[K];
      R_POLY.C[J]:=  R_POLY.C[J]   +   H[2*K-1]*R[K];
     END;
      L_POLY.C[J]:=L_POLY.C[J]*FAC;     B_POLY.C[J]:=B_POLY.C[J]*FAC;
      R_POLY.C[J]:=R_POLY.C[J]*FAC;
     END;
  END;
(*----------------------------------------------------------------- *)
(* T_FIT_MOON: Вычисление разложения Чебышева для координат Луны*)
(*              (ряды для RA,DEC и R).                              *)
```

```
(*                                                              *)
(*    TA: начальный момент интервала разложения                 *)
(*        (в юл. столетиях с J2000)                             *)
(*    TB: конечный момент интервала разложения                  *)
(*        (TB < TA + 1 месяц)                                   *)
(*    N : порядок полинома                                      *)
(*    RA_POLY,DE_POLY,R_POLY: полином Чебышева для RA,DEC,R     *)
(*--------------------------------------------------------------*)
PROCEDURE T_FIT_MOON ( TA,TB: REAL; N: INTEGER;
                       VAR RA_POLY,DE_POLY,R_POLY: TPOLYNOM);
  BEGIN
    T_FIT_LBR(MOONEQU,TA,TB,N,RA_POLY,DE_POLY,R_POLY);
  END;
(*--------------------------------------------------------------*)
```

При вычислении разложения по полиномам Чебышева совершенно не нужно вычислять полиномы в явном виде. Алгоритм, предложенный Кленшоу (Clenshaw), использует рекурсивное соотношение (6.10) и дает следующее правило вычисления разложения Чебышева (6.12) порядка n с коэффициентами (c_0, c_1,..., c_n):

* Принимаем $f_{n+1} = 0$ и $f_{n+2} = 0$.
* Вычисляем для нормированного аргумента $\hat{x}$ последовательность

$$f_i = 2\hat{x}\, f_{i+1} - f_{i+2} + c_i \qquad \text{при } i = n,\ n-1, \ldots, 0.$$

* Искомое значение функции равно

$$f(x) = (f_0 - f_2)/2 = \hat{x} f_1 - f_2 - c_0/2\,.$$

Функция T_EVAL по этой схеме оценивает ряд T.

```
(*--------------------------------------------------------------*)
(* T_EVAL: Вычисление в интервале [F.A,F.B] значений функции,   *)
(*         разложенной по полиномам Чебышева                    *)
(*         наивысшего порядка F.M                               *)
(*      F: полином Чебышева                                     *)
(*      X: аргумент                                             *)
(*--------------------------------------------------------------*)
 FUNCTION T_EVAL(F: TPOLYNOM; X: REAL):REAL;
  VAR F1,F2,OLD_F1,XX,XX2          : REAL;
      I                            : INTEGER;
  BEGIN
    IF ( (X<F.A) OR (F.B<X) ) THEN
      BEGIN WRITELN(' T_EVAL : x не в [a,b]'); END;
    F1   := 0.0;  F2:= 0.0;
    XX   := (2.0*X-F.A-F.B)/(F.B-F.A);      XX2 := 2.0*XX;
  FOR I := F.M DOWNTO 1 DO
    BEGIN OLD_F1:= F1; F1  := XX2*F1-F2+F.C[I];   F2 := OLD_F1;   END;
     T_EVAL:=   XX*F1-F2+0.5*F.C[0]
  END;
(*--------------------------------------------------------------*)
```

6.4. Программа LUNA

В программе LUNA разные подпрограммы, представленные в этой главе, собраны для небольшого конкретного приложения. По этой программе можно вычислить эфемериды Луны, т. е. таблицы положений Луны в том виде, как они приводятся в ежегодниках. В результате вычислений выдаются видимые (приведенные к равноденствию даты) экваториальные координаты Луны, ее расстояние в единицах земного радиуса и ее горизонтальный параллакс. LUNA при необходимости разлагает координаты Луны в ряд Чебышева в интервале 10 суток. Это разложение вычисляется легко и быстро. Для построения достаточно точного приближения вычисляются 13 положений Луны. Если координаты Луны нужны с шагом в одни сутки или более, построение ряда требует излишнего счета. Напротив, используя этот метод, эфемериды с малым шагом (необходимые для навигационных задач) можно вычислить со значительной экономией усилий.

```
(*-----------------------------------------------------------*)
(*                          LUNA                             *)
(*                     Эфемериды Луны                        *)
(*                   Версия 28.09.1988                       *)
(*-----------------------------------------------------------*)

PROGRAM LUNA(INPUT,OUTPUT);

  CONST MAX_TP_DEG=13;                    (* порядок разложения*)
     T_OVERLAP   = 3.42E-6;         (* 3h в юлианских столетиях  *)
     T_DEVELOP   = 2.737850787E-4;  (* 10d в юлианских столетиях *)

TYPE TPOLYNOM  = RECORD                   (* полином Чебышева *)
              M  : INTEGER;                      (*   степень  *)
              A,B: REAL;                         (* интервал   *)
              C  : ARRAY [0..MAX_TP_DEG] OF REAL; (*   коэффиц.*)
         END;

   VAR RA,DE,R,PAR, MODJD,HOUR: REAL;
       T,DT,T_START,T_END,TA,TB: REAL;
       DAY,MONTH,YEAR,NLINE    : INTEGER;
       RA_POLY,DE_POLY,R_POLY: TPOLYNOM;

(*-----------------------------------------------------------*)
(* Сюда вводятся следующие подпрограммы, в указанной         *)
(* последовательности:                                       *)
(* SN, CS, ASN, ATN, ATN2, CART, POLAR, GMS, T_EVAL, T_FIT_LBR *)
(*   MJD, CALDAT, ECLEQU, NUTEQU                             *)
(*   MOON, MOONEQU, T_FIT_MOON                               *)
(*-----------------------------------------------------------*)

(*-----------------------------------------------------------*)
(* GETEPH: Ввод временного интервала вычисления эфемерид     *)
(*-----------------------------------------------------------*)
```

```
PROCEDURE GETEPH(VAR T1,DT,T2:REAL);
  VAR YEAR,MONTH,DAY:INTEGER;
      HOUR           : REAL;
  BEGIN
   WRITELN;
   WRITELN('                LUNA: Эфемериды Луны                    ');
   WRITELN('                   Версия 28.09.88                      ');
   WRITELN('   (c) 1988 Thomas Pfleger, Oliver Montenbruck          ');
   WRITELN;
   WRITELN(' Начало и конец эфемерид:                               ');
   WRITELN;
   WRITE (' первая дата вычислений (DD MM YYYY HH.HHH)             ');
   READLN(DAY,MONTH,YEAR,HOUR);
   T1.= ( MJD(DAY,MONTH,YEAR,HOUR) - 51544.5 ) / 36525.0;
   WRITE (' последняя дата вычислений (DD MM YYYY HH.HHH)          ');
   READLN(DAY,MONTH,YEAR,HOUR);
   T2.= ( MJD(DAY,MONTH,YEAR,HOUR) - 51544.5 ) / 36525.0;
   WRITE ('  шаг  (DD HH.HH)                                       ');
   READLN (DAY,HOUR);
   DT:= ( DAY + HOUR/24.0 ) / 36525.0;
  END;
(*-----------------------------------------------------------------*)
(* WRTLBRP: форматированный вывод                                  *)
(*-----------------------------------------------------------------*)
PROCEDURE WRTLBRP (L,B,R,P:REAL),
  VAR H,M: INTEGER;
      S · REAL;
  BEGIN
   GMS(L,H,M,S);   WRITE  (H:5,M:3,S:5:1);
   GMS(B,H,M,S);   WRITE (H:5,M·3,S:5:1),    WRITE  (R:10:3);
   GMS(P,H,M,S);   IF (H>0) THEN M:=M+60;   WRITELN  (M:6,S:6:2);
  END;
(*-----------------------------------------------------------------*)

BEGIN   (* Основная программа   *)

  GETEPH(T_START,DT,T_END);  (* считывание временного интервала   *)

  WRITELN;
  WRITE  ('   Дата     ET        Ra          Dec        расстояние  ');
  WRITELN('   параллакс');
  WRITE  ('            h         h   m   s      o  ''  "   радиус  Земли')·
  WRITELN('    ''  "                                                    ');

  T:= T_START;       TB:= T_START;     NLINE:= 0;

  WHILE (T<=T_END) DO
```

6.4. Программа LUNA

В программе LUNA разные подпрограммы, представленные в этой главе, собраны для небольшого конкретного приложения. По этой программе можно вычислить эфемериды Луны, т. е. таблицы положений Луны в том виде, как они приводятся в ежегодниках. В результате вычислений выдаются видимые (приведенные к равноденствию даты) экваториальные координаты Луны, ее расстояние в единицах земного радиуса и ее горизонтальный параллакс. LUNA при необходимости разлагает координаты Луны в ряд Чебышева в интервале 10 суток. Это разложение вычисляется легко и быстро. Для построения достаточно точного приближения вычисляются 13 положений Луны. Если координаты Луны нужны с шагом в одни сутки или более, построение ряда требует излишнего счета. Напротив, используя этот метод, эфемериды с малым шагом (необходимые для навигационных задач) можно вычислить со значительной экономией усилий.

```
(*-----------------------------------------------------------------*)
(*                            LUNA                                  *)
(*                       Эфемериды Луны                             *)
(*                     Версия 28.09.1988                            *)
(*-----------------------------------------------------------------*)

PROGRAM LUNA(INPUT,OUTPUT);
  CONST MAX_TP_DEG=13;                          (* порядок разложения*)
     T_OVERLAP   = 3.42E-6;             (* 3h в юлианских столетиях  *)
     T_DEVELOP   = 2.737850787E-4;      (* 10d в юлианских столетиях *)
TYPE TPOLYNOM    = RECORD                       (* полином Чебышева *)
              M  · INTEGER;                     (*     степень      *)
              A,B: REAL;                        (*  интервал        *)
              C  : ARRAY [0..MAX_TP_DEG] OF REAL; (*    коэффиц.*)
         END;
  VAR RA,DE,R,PAR, MODJD,HOUR:  REAL;
      T,DT,T_START,T_END,TA,TB: REAL;
      DAY,MONTH,YEAR,NLINE  :  INTEGER;
      RA_POLY,DE_POLY,R_POLY:  TPOLYNOM;
(*-----------------------------------------------------------------*)
(* Сюда вводятся следующие подпрограммы, в указанной                *)
(* последовательности:                                              *)
(* SN, CS, ASN, ATN, ATN2, CART, POLAR, GMS, T_EVAL, T_FIT_LBR      *)
(*   MJD, CALDAT, ECLEQU, NUTEQU                                    *)
(*   MOON, MOONEQU, T_FIT_MOON                                      *)
(*-----------------------------------------------------------------*)

(*-----------------------------------------------------------------*)
(* GETEPH: Ввод временного интервала вычисления эфемерид            *)
(*-----------------------------------------------------------------*)
```

```
PROCEDURE GETEPH(VAR T1,DT,T2:REAL);
  VAR YEAR,MONTH,DAY:INTEGER;
      HOUR            : REAL;
  BEGIN
   WRITELN;
   WRITELN('                LUNA: Эфемериды Луны                  ');
   WRITELN('                   Версия 28.09.88                    ');
   WRITELN('   (c) 1988 Thomas Pfleger, Oliver Montenbruck        ');
   WRITELN;
   WRITELN(' Начало и конец эфемерид:                             ');
   WRITELN;
   WRITE (' первая дата вычислений (DD MM YYYY HH.HHH)           ');
   READLN(DAY,MONTH,YEAR,HOUR);
   T1 =  ( MJD(DAY,MONTH,YEAR,HOUR) - 51544.5 ) / 36525.0;
   WRITE (' последняя дата вычислений (DD MM YYYY HH.HHH)        ');
   READLN(DAY,MONTH,YEAR,HOUR);
   T2:=  ( MJD(DAY,MONTH,YEAR,HOUR) - 51544.5 ) / 36525.0;
   WRITE ('  шаг  (DD HH HH)                                     ');
   READLN (DAY,HOUR);
   DT = ( DAY + HOUR/24.0 ) / 36525.0;
  END,
(*-------------------------------------------------------------*)
(* WRTLBRP: форматированный вывод                              *)
(*-------------------------------------------------------------*)
PROCEDURE WRTLBRP (L,B,R,P:REAL);
  VAR H,M·INTEGER;
      S  : REAL;
  BEGIN
   GMS(L,H,M,S);   WRITE   (H:5,M:3,S:5:1);
   GMS(B,H,M,S);   WRITE (H:5,M 3,S:5:1);    WRITE   (R:10:3);
   GMS(P,H,M,S);   IF (H>0) THEN M:=M+60;   WRITELN  (M:6,S:6:2);
  END;
(*-------------------------------------------------------------*)

BEGIN    (* Основная программа    *)

  GETEPH(T_START,DT,T_END);  (* считывание временного интервала  *)

  WRITELN;
  WRITE  ('    Дата     ET       Ra          Dec        расстояние   ');
  WRITELN('   параллакс');
  WRITE  ('           h       h  m   s       o  ''  "     радиус  Земли')·
  WRITELN('     ''  "                                                    ');

  T:= T_START;       TB:= T_START;    NLINE:= 0;

  WHILE (T<=T_END) DO
```

```
      BEGIN

    IF (T>TB-T_OVERLAP) THEN  (* новое разложение координат          *)
          BEGIN
            TA:= T-T_OVERLAP;  TB:= T+T_DEVELOP+T_OVERLAP;
            T_FIT_MOON (TA,TB,MAX_TP_DEG,RA_POLY,DE_POLY,R_POLY);
          END;

        (* дата            *)
   MODJD:= T*36525.0 + 51544.5;  CALDAT (MODJD,DAY,MONTH,YEAR,HOUR);
        WRITE   (DAY:3,MONTH:3,YEAR:5,HOUR:5:1);

        (* координаты     *)
      RA := T_EVAL(RA_POLY,T)/15.0;   IF RA<0.0 THEN RA:= RA + 24.0;
      DE := T_EVAL(DE_POLY,T);
      R  := T_EVAL(R_POLY, T);
      PAR:= ASN(1 0/R);

        (* вывод координат        *)
        WRTLBRP (RA,DE,R,PAR);
        NLINE:= NLINE + 1;
        IF (NLINE MOD 5) =0 THEN WRITELN;

        T:= T + DT;

     END;
  END.
  (*-------------------------------------------------------------------*)
```

В начале работы программа LUNA вычисляет разложение координат Луны в ряд, что занимает несколько секунд. Затем координаты выдаются в быстрой последовательности, пока не понадобится (для последующего временного интервала) обновить ряд.

В качестве примера применения программы, вычислим эфемериды Луны для января 1989 г. с шагом в двое суток. LUNA требует ввода только граничных дат в форме «день, месяц, год и часы с десятичными долями». Все временные данные вводятся в единицах эфемеридного времени (как и случае вычисления положения планет). Вводимые данные в дальнейшем приведены курсивом.

LUNA: Эфемериды Луны
Версия 28.09.88
(c) 1988 Thomas Pfleger, Oliver Montenbruck

Начало и конец эфемерид:
первая дата вычислений (DD MM YYYY HH.HHH) *1 1 1989 0.*
последняя дата вычислений (DD MM YYYY HH.HHH) *31 1 1989 0.*
шаг (DD HH.HH) *2 0.*

Дата	ET	Ra			Dec			Расстояние	Параллакс	
	h	h	m	s	°		″	радиус Земли	′	″
1 1 1989	0.0	13	5	26.2	-10	42	58.7	63.053	54	31.41
3 1 1989	0 0	14	38	15 3	-20	23	19.2	61.961	55	29.30
5 1 1989	0.0	16	26	11 9	-26	51	42.9	60.407	56	54.71
7 1 1989	0.0	18	27	52 8	-27	43	3.3	58 861	58	24.44
9 1 1989	0.0	20	29	55.2	-21	44	30.5	57.789	59	29.44
11 1 1989	0.0	22	21	11.9	-10	29	51.2	57.442	59	51.00
13 1 1989	0.0	0	3	44.1	2	54	49.4	57.757	59	31.44
15 1 1989	0.0	1	46	21.7	15	30	4.5	58.481	58	47.18
17 1 1989	0.0	3	36	54.1	24	39	14.7	59.389	57	53.29
19 1 1989	0.0	5	35	18.6	28	13	6.1	60.375	56	56.57
21 1 1989	0.0	7	31	1.7	25	29	44.7	61.406	55	59.16
23 1 1989	0.0	9	14	3 9	17	51	26.1	62.408	55	5.27
25 1 1989	0 0	10	44	27.6	7	31	36.2	63.194	54	24.13
27 1 1989	0.0	12	8	35.2	-3	38	59.4	63 512	54	7.79
29 1 1989	0.0	13	34	26.4	-14	17	56.9	63.141	54	26.88
31 1 1989	0.0	15	10	2.7	-23	1	46.5	62.014	55	26.26

В этой таблице обращают на себя внимание высокие склонения Луны 7 и 19 января 1989 г. В этом году линия узлов орбиты Луны расположена таким образом, что наклон эклиптики (≈ 23,5°) и наклон орбиты Луны (≈ 5,1°) складываются и склонение может превосходить 28°. В определенные моменты времени в этом году Луна находится особенно высоко или низко на небе; азимуты точек ее восхода и захода также могут достигать экстремальных значений.

7. ЗАТМЕНИЯ СОЛНЦА

Примерно через каждые тридцать суток Луна во время новолуния поворачивается к Земле неосвещенной стороной. Для наблюдателя, смотрящего с севера на эклиптику, Солнце, Луна и Земля расположены в это время на одной прямой. Однако тень Луны очень редко падает на Землю. Из-за наклона орбиты Луна во время новолуния в большинстве случаев находится над или под плоскостью орбиты Земли, и тень не попадает на Землю. Луна пересекает эклиптику только в течение двух дней в месяц, и если одна из этих дат совпадает с новолунием, то Солнце, Луна и Земля оказываются на одной прямой и тень Луны падает на часть земной поверхности. В общем такие солнечные затмения происходят два раза в год. Каждый, кто находится внутри 100-км тени Луны, видит полное затмение Солнца. Это относительно небольшая область, так что свидетелями данного полного солнечного затмения становятся лишь немногие.

Если бы орбита Луны была жестко ориентирована в пространстве, то все затмения происходили бы в два определенных месяца. Из-за движения линии узлов даты затмений перемещаются за год в среднем на три недели. Таким образом, 18,6-летний период обращения лунных узлов прямо отражается в ритме солнечных затмений (рис.7.1).

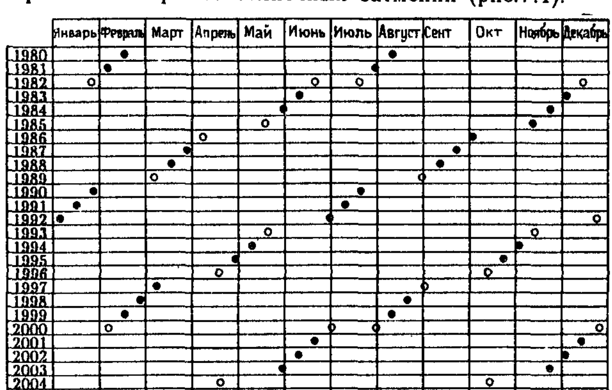


Рис.7.1. Солнечные затмения в промежутке 1980—2005 гг. (•—полное/кольцеобразное, ∘—частное).

7.1. Моменты новолуний

Чтобы выяснить ситуацию с солнечными затмениями в течение года, необходимо в первую очередь определить даты новолуний на каждый месяц и исследовать положение Луны в эти моменты относительно эклиптики. Новолуние — это момент, когда эклиптические долготы Солнца и Луны ($\lambda_\odot$ и λ_M соответственно) совпадают, т. е. разность $\lambda_M - \lambda_\odot$ обращается в нуль. Эта разность в свою очередь есть сумма разности D средних долгот и разности периодических возмущений:

$$\lambda_M - \lambda_\odot = D + (\Delta\lambda_M - \Delta\lambda_\odot) .$$

Согласно (6.5), элонгация D равна

$$D = D_0 + D_1 \cdot T = 297{,}85027^\circ + 445267{,}11135^\circ\ T,$$

где

$$T = (\mathrm{JD} - 2451545)/36525 .$$

D_1 показывает, насколько D меняется в течение столетия. Интервал времени между двумя новолуниями, т. е. время, за которое D меняется на 360°, составляет в среднем 29,53 сут. Это дает возможность очень просто вычислить *приблизительные моменты новолуний на год.*

Периодические возмущения солнечной и лунной орбиты очень мало меняются за время Δt, так что $\lambda_M - \lambda_\odot$ за это время изменяется, в сущности, на величину $D_1 \cdot \Delta t/36525^{\mathrm{d}}$. Приближенный момент новолуния t_0 можно поэтому уточнить:

$$t_1 = t_0 - \frac{D\ (t_0) - (\Delta\lambda_M\,(t_0) - \Delta\lambda_\odot(t_0))}{D_1} \cdot 36525^{\mathrm{d}}.$$

Повторение этой операции позволяет вычислить момент новолуния весьма точно. Величины $\Delta\lambda_M$ и $\Delta\lambda_\odot$ задаются небольшими рядами, аргументами которых служат средние долготы и аномалии Солнца и Луны ((6.2)—(6.5)):

$$\begin{aligned} \Delta\lambda_M &= +22640''\ \sin\,(l) - 4586''\ \sin\,(l - 2D) + 2370''\ \sin\,(2D) + \\ &\quad +769''\ \sin\,(2l) - 668''\ \sin\,(l') + \ldots, \\ \Delta\lambda_\odot &= +6893''\ \sin\,(l') + 72''\ \sin\,(2l') . \end{aligned}$$

Подобным разложением в ряд можно представить и эклиптическую широту Луны:

$$\beta_M \approx +18520''\ \sin\,(F + \Delta\lambda) - 526''\ \sin\,(F - 2D).$$

Объяснения к этой формуле даны в гл.6.

Все эти действия объединены в программе NEWMOON. С ее помощью, задавая нужный год, определяют отдельные моменты новолуний и соответствующие им эклиптические широты Луны. Полученная информация позволяет достаточно точно предсказывать наступление солнечных затмений.

Вследствие наклона орбиты Луны к эклиптике β в течение года колеблется между –5 и +5°. В общем, солнечные затмения происходят в два новолуния, которым соответствует наименьшая эклиптическая широта. В некоторые годы (например, 1982 или 2000) вместо двух полных затмений могут случиться четыре частных затмения. Наибольшее значение эклиптической широты, при котором возможно полное затмение, равняется примерно одному градусу. Частные солнечные затмения могут происходить до $\beta \approx 1{,}5^\circ$. Кроме того, мы можем уже приблизительно определить, где будет видно определенное затмение. Если Луна находится к северу от эклиптики ($\beta > 0$), то область тени будет лежать в Северном полушарии.

```
(*-----------------------------------------------------------------*)
(*                          NEWMOON                                 *)
(*  Вычисление момента новолуния и эклиптической широты Луны        *)
(*                       Версия 12.11.88                            *)
(*-----------------------------------------------------------------*)
PROGRAM NEWMOON (INPUT,OUTPUT);
CONST D1= +1236.853086;  (* Изменения разности средних долгот     *)
                         (* Луны и Солнца                          *)
                         (* dD/dT = 1236.85 оборот/столетие        *)
      D0 =    +0.827361; (* Разность ср. долгот Луны и Солнца      *)
                         (* для J2000 (в оборотах)                 *)

VAR   DAY,MONTH,YEAR,YEAR_CALC:INTEGER;
      HOUR                    : REAL;
      LUNATION_0,LUNATION_I  : INTEGER;
      T_NEW_MOON,MJD_NEW_MOON: REAL;
      B_MOON                 : REAL;
(*-----------------------------------------------------------------*)
(*  Сюда вводятся подпрограммы MJD и CALDAT                        *)
(*-----------------------------------------------------------------*)
(*-----------------------------------------------------------------*)
(* IMPROVE: уточнение приближения T времени новолуния              *)
(* и определение приближенной эклиптической широты Луны B          *)
(* ( T в юл. столетиях с J2000, T = (JD-2451545)/36525 )           *)
(*-----------------------------------------------------------------*)
PROCEDURE IMPROVE ( VAR T,B: REAL);
  CONST P2 =6.283185307;       (* 2*pi                             *)
        ARC=206264 8062;       (* секунд дуги в радиане            *)
  VAR   L,LS,D,F,DLM,DLS,DLAMBDA:REAL;
(* TRUNC при T<-24 заменить на LONG_TRUNC или INT !                *)
  FUNCTION FRAC(X:REAL):REAL;
    BEGIN X:=X-TRUNC(X), IF (X<0) THEN X:=X+1; FRAC:=X   END;
  BEGIN
    (*  L,LS,D,F-средние элементы орбиты Луны                       *)
 L :=  P2*FRAC(0 374897+1325 552410*T);   (* ср. аномалия Луны      *)
 LS:=  P2*FRAC(0 993133+  99.997361*T);   (* ср. аномалия Солнца    *)
```

```
  D :=  P2*(FRAC(0.5+D0+D1*T)-0.5); (*ср.разн.долгот Луна—Солнце     *)
  F :=  P2*FRAC(0.259086+1342.227825*T);(* расстояние узла          *)
  (* периодические возмущения долгот Солнца и Луны (сек. дуги)      *)
  DLM:= +22640*SIN(L)      - 4586*SIN(L-2*D) + 2370*SIN(2*D)
        +769*SIN(2*L)      - 668*SIN(LS) - 412*SIN(2*F)
        -212*SIN(2*L-2*D)  - 206*SIN(L+LS-2*D) + 192*SIN(L+2*D)
        -165*SIN(LS-2*D)   - 125*SIN(D) - 110*SIN(L+LS)
        +148*SIN(L-LS)     - 55*SIN(2*F-2*D);
  DLS:= +6893*SIN(LS)      + 72*SIN(2*LS);
  (* разность истинных долгот Луны и Солнца (в оборотах)            *)
  DLAMBDA:= D / P2 + ( DLM - DLS) / 1296000.0;
  (* поправка момента новолуния                                     *)
  T :=  T - DLAMBDA / D1;
  (* эклиптическая широта B Луны (град.)                            *)
  B := ( + 18520.0*SIN(F+DLM/ARC) - 526*SIN(F-2*D) ) / 3600.0;
END;
(*----------------------------------------------------------------  *)

BEGIN (* Основная программа       *)

  WRITELN;
  WRITELN('NEWMOON:Моменты новолуния и эклиптическая широта Луны');
  WRITELN ('                    Версия 12.11.88                    ');
  WRITELN ('   (c) 1988 Thomas Pfleger, Oliver Montenbruck         ');
  WRITELN;
  WRITE  (' Моменты новолуния для года ');   READLN(YEAR_CALC);
  WRITELN;
  WRITELN (' Дата':16,'UT':7,'широта':11);        WRITELN;
  WRITELN (' h':23,'o':9);

  LUNATION_0    :=  TRUNC( D1 * (YEAR_CALC-2000)/100 );
  FOR LUNATION_I:=  LUNATION_0 TO LUNATION_0 + 13 DO
    BEGIN
      T_NEW_MOON:= ( LUNATION_I - D0 ) / D1;
      IMPROVE ( T_NEW_MOON, B_MOON );
      IMPROVE ( T_NEW_MOON, B_MOON );
      MJD_NEW_MOON := 36525.0*T_NEW_MOON + 51544.5;
      CALDAT ( MJD_NEW_MOON, DAY,MONTH,YEAR,HOUR );
      IF YEAR=YEAR_CALC THEN
         WRITELN(DAY:10,MONTH:3,YEAR:5,HOUR:6:1,B_MOON:9:1)
    END;
   WRITELN;

END.
(*----------------------------------------------------------------  *)
```

7.2. Геометрия затмений

Лучи Солнца образуют две конусообразные области, которые называются *тенью* и *полутенью* (рис.7.2). Для наблюдателя, находящегося в области **тени**, Луна кажется больше Солнца и полностью его закрывает. Вершина конуса расположена на расстоянии около 375 000 км от Луны. Далее располагается область, где затмение выглядит кольцеобразным. Отсюда Луна кажется темным диском, расположенным перед Солнцем, но не закрывающим его полностью. Часть Солнца при этом остается видимой как кольцо вокруг Луны. Примером такого затмения может служить кольцеобразное затмение 29 апреля 1976 г., которое можно было наблюдать на территории Средиземноморья.

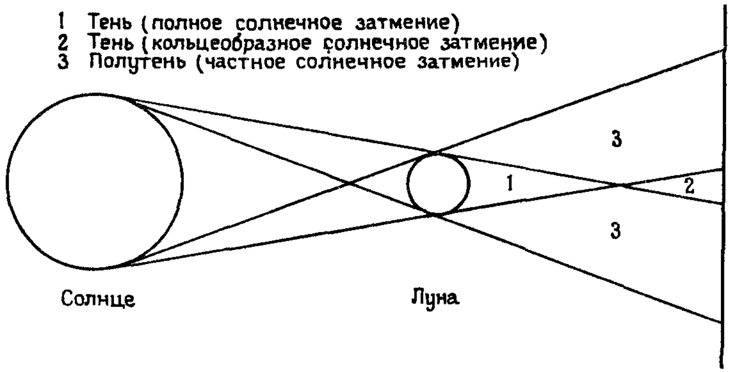


Рис.7.2. Тень и полутень.

В полутени Солнце всегда затемнено только частично и выглядит как более или менее широкий серп. Фаза затмения тем больше, чем ближе наблюдатель к конусу тени.

Расстояние Луны от Земли составляет в среднем 380 000 км и меняется из-за эксцентриситета лунной орбиты более чем на 25 000 км. Поэтому во время затмения Земля всегда находится около вершины конуса тени. Даже в перигее, когда Луна и Земля наиболее близки друг к другу, диаметр тени на поверхности Земли редко превышает 200 км. В апогее затмение бывает кольцеобразным. Случаются промежуточные ситуации, когда в течение некоторого времени затмение полное, а затем переходит в кольцеобразное. Таким было затмение 29 марта 1987 г

В плоскости, перпендикулярной линии, соединяющей Солнце и Луну, на расстоянии s за Луной диаметры тени d и полутени D составляют:

$$d\,(s) = D_\odot \left[\frac{s}{r_{\odot M}} \right] - D_M \left[1 + \frac{s}{r_{\odot M}} \right] \text{ и}$$

$$D\,(s) = D_\odot \left[\frac{s}{r_{\odot M}} \right] + D_M \left[1 + \frac{s}{r_{\odot M}} \right]. \tag{7.1}$$

Здесь $D_\odot$ и D_M—диаметры Солнца и Луны:

$$D_\odot = 1392000 \text{ км} = 218{,}25\ R_\oplus,$$

$$D_M = 3476 \text{ км} = 0{,}5450\ R_\oplus,$$

$r_{\odot M}$—расстояние между Солнцем и Землей.

Вывод формул для диаметра тени проиллюстрирован на рис.7.3. Здесь f—половина угла раствора конуса тени и l—расстояние вершины тени от центра Луны Если $R_\odot$ и R_M—соответственно радиусы Солнца и Луны, получим

$$\sin(f) \cdot l = R_M \quad \text{и} \quad \sin(f) \cdot (l + r_{\odot M}) = R_\odot,$$

или в преобразованном виде

$$\sin(f) = \frac{R_\odot - R_M}{r_{\odot M}} \quad \text{и} \quad l = \frac{R_M \cdot r_{\odot M}}{R_\odot - R_M}.$$

Диаметр тени на расстоянии s от Луны будет

$$d = 2\,(s - l) \cdot \operatorname{tg}(f) \approx 2\,(s - l) \cdot \sin(f) =$$

$$= 2R_\odot \left[\frac{s}{r_{\odot M}} \right] - 2R_M \left[1 + \frac{s}{r_{\odot M}} \right].$$

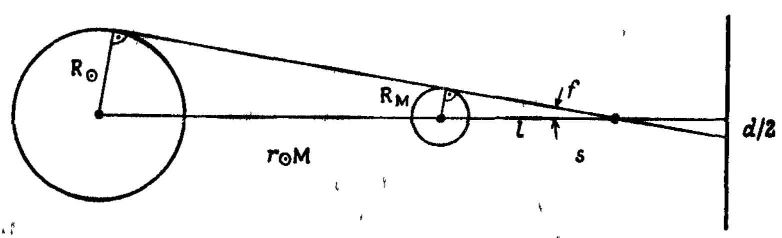


Рис.7.3. Вычисление диаметра тени.

Точно так же выводится уравнение диаметра полутени Знак d в этом уравнении выбирается таким образом, чтобы в области тени, т е при *полном затмении, d было* ***отрицательным***, *а при частном затмении положительным* Так как диаметр полутени D вблизи Земли приблизительно равен половине размера самой Земли, то диаметр тени d колеблется между $-0{,}04R_\oplus$ (250 км) и $+0{,}06R_\oplus$ (350 км)

Положение тени на Земле определяют, проводя прямую через центры Солнца и Луны и продолжая ее до пересечения с поверхностью Земли (рис 7 4) Направление этой прямой описывается вектором единичной длины

$$\mathbf{e} = \frac{\mathbf{r}_M - \mathbf{r}_\odot}{|\mathbf{r}_M - \mathbf{r}_\odot|}, \tag{7 2}$$

рассчитываемым на основе координат $\mathbf{r}_\odot$ и $\mathbf{r}_M$ Солнца и Луны. Для точки $\mathbf{r}$, находящейся в пределах тени, имеем

$$\mathbf{r} = \mathbf{r}_M + s\,\mathbf{e}, \tag{7 3}$$

или по компонентам

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x_M \\ y_M \\ z_M \end{bmatrix} + s \cdot \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix}$$

Здесь s—расстояние между точкой тени и центром Луны Так как конец $\mathbf{r}$ лежит на поверхности Земли, считая пока Землю идеальным шаром радиусом $R_\oplus$, получаем

$$\mathbf{r}^2 = x^2 + y^2 + z^2 = R_\oplus^2 \tag{7.4}$$

Подставив, получим квадратное уравнение

$$s^2 + 2\,(\mathbf{r}_M\mathbf{e})\,s + (r_M^2 - R_\oplus^2) = 0$$

Оно имеет два решения

$$s = \begin{cases} s_0 - \sqrt{\Delta} & \text{дневная сторона Земли,} \\ s_0 + \sqrt{\Delta} & \text{ночная сторона Земли,} \end{cases}$$

где

$$\begin{aligned} s_0 &= -\mathbf{r}_M\mathbf{e} = -\,(x_M e_x + y_M e_y + z_M e_z), \\ \Delta &= s_0^2 + R_\oplus^2 - r_M^2 \end{aligned} \tag{7.5}$$

s_0—расстояние Луны от так называемой фундаментальной плоскости, перпендикулярной оси тени и проходящей через центр Земли Искомая точка пересечения оси тени с земным шаром расположена на расстоянии

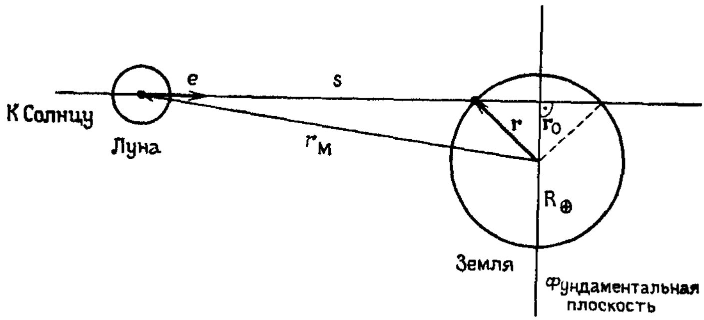


Рис.7.4. Пересечение оси тени с Землей

$\sqrt{\Delta}$ от этой плоскости на стороне, обращенной к Луне и Солнцу Вторая точка пересечения лежит на ночной стороне Земли и для затмения несущественна Объединяя приведенные выражения, получаем для координат тени на поверхности Земли

$$\mathbf{r} = \mathbf{r}_M + (s_0 - \sqrt{\Delta}) \cdot \mathbf{e} \tag{7 6}$$

Естественно, при этом предполагается, что дискриминант Δ положителен, иначе ось тени пройдет мимо Земли Тем не менее затмение Солнца начинается до этой центральной фазы Как только конус полутени коснется Земли, в некоторых областях Земли можно наблюдать частное затмение Если

$$r_0 = \sqrt{r_M^2 - s_0^2}$$

—расстояние оси тени от центра Земли, а d_0 и D_0—диаметры тени и полутени на фундаментальной плоскости, то отдельные фазы затмения можно в принципе различать по следующим признакам

$R_\oplus + D_0/2 < r_0$	затмения нет,
$R_\oplus + \lvert d_0\rvert/2 < r_0 < R_\oplus + D_0/2$	частное затмение,
$R_\oplus < r_0 < R_\oplus + \lvert d_0\rvert/2$	нецентральная фаза,
$r_0 < R_\oplus$	центральная фаза

Во время нецентральной фазы конус тени так касается Земли, что полное или кольцеобразное затмение можно наблюдать только на малой части поверхности Земли Сама ось тени в этой фазе вообще не пересекается с Землей В этом случае на Земле нет такого места, откуда Луна была бы видна точно перед Солнцем и центры их совпадали бы

Все приведенные выше рассуждения исходят из предположения, что Земля имеет форму идеального шара. Для учета небольшой сплюснутости Земли при вычислении затмений нужны некоторые поправки. Предположим, что поверхность Земли получается из сферы радиусом $R_{\oplus}$, сплюснутой вдоль земной оси в $1 - f = 0{,}996647$ раз. Каждая точка (x, y, z) такого эллипсоида вращения удовлетворяет уравнению

$$r^2 = x^2 + y^2 + z^2/(1 - f)^2 = R_{\oplus}^2 , \tag{7.7}$$

которое получается из уравнения (7.4) заменой z на $z/(1 - f)$. Ось z используемой системы координат должна быть ориентирована параллельно земной оси (экваториальные координаты).

Исходя из факта, что эллипсоид при соответствующем растяжении превращается в шар, нетрудно, несмотря на сплюснутость, правильно определить положение тени. Для этого z-координаты Солнца и Луны умножают на коэффициент $1/(1 - f)$, проводят прямую через измененные положения и продолжают ее до пересечения с шаром, радиус которого равен радиусу Земли. Вычисленная таким способом точка пересечения отличается от z координаты точки пересечения истинной тени с поверхностью Земли только коэффициентом $1/(1 - f)$. При этом полученные выше зависимости могут использоваться без изменения.

7.3. Географические координаты и сплюснутость Земли

Для описания положения точки на поверхности Земли с помощью координат обычно используют географическую долготу λ и географическую широту φ. Они весьма сходны с экваториальными координатами — прямым восхождением α и склонением δ, применяемыми в астрономии. Обе системы координат имеют оси, параллельные плоскости экватора и оси вращения Земли

Пересчет склонения в географическую широту и обратно был бы излишним, если бы Земля действительно была шаром; в этом случае обе координаты были бы идентичны Но, поскольку Земля является сплюснутым — пусть совсем ненамного — эллипсоидом вращения, нужно прежде всего немного уточнить понятие географической широты φ. Она определяет угол наклона локального горизонта любого места к земной оси, как показано на рис 7 5 Сечение, проходящее через Северный и Южный полюсы Земли, имеет форму эллипса, а горизонт является касательной к нему. Очевидно, что величина φ не идентична ни геоцентрической широте φ', ни склонению δ, которые представляют собой угол между вектором положения и плоскостью экватора Величины φ и φ' связаны соотношениями

$$\begin{aligned} r \cos(\varphi') &= \frac{R_{\oplus}}{\sqrt{1 - e^2 \sin^2\varphi}} \cos\varphi , \\ r \sin(\varphi') &= \frac{(1 - e^2)\, R_{\oplus}}{\sqrt{1 - e^2 \sin^2\varphi}} \sin\varphi , \\ \operatorname{tg}(\varphi') &= (1 - e^2) \operatorname{tg}\varphi . \end{aligned} \tag{7.8}$$

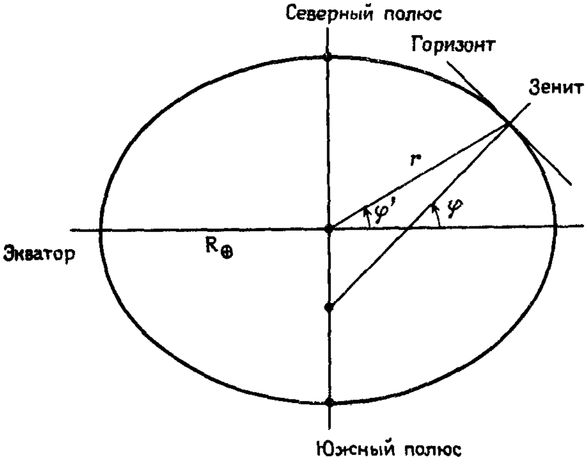


Рис.7.5. К сплюснутости Земли.

Здесь $R_\oplus$ = 6378,14 км — экваториальный радиус Земли, а e — эксцентриситет земного эллипсоида. Под действием центробежной силы, связанной с вращением, Земля приняла такую свою форму, что экваториальный радиус примерно на 20 км больше расстояния R_{pol} между центром Земли и полюсами Отношение этих величин и определяет сплюснутость

$$f = \frac{R_\oplus - R_{pol}}{R} = \frac{21,385 \text{ км}}{6378,14 \text{ км}} = \frac{1}{298,257} = 0,003353. \qquad (7.9)$$

Эксцентриситет Земли вычисляется по формуле

$$e = \sqrt{1 - (1 - f^2)} = \sqrt{2f - f^2}\,. \qquad (7.10)$$

Вместо строгого уравнения (7 8) для вычисления географической широты можно использовать достаточно точное приближение

$$\varphi = \varphi' + 0,1924^\circ \sin(2\varphi') \qquad (7.11)$$

Из него хорошо видно, что различие между φ и φ' максимально для средних широт, где составляет примерно 12'. На полюсе и экваторе разница исчезает

Наряду с географической широтой для однозначного описания места на Земле в качестве второй координаты используют географическую долготу Она в основном соответствует прямому восхождению в экваториальной системе координат Однако исчисление прямого восхождения ведется от жестко фиксированной в пространстве точки весеннего равноденствия, а географическая долгота исчисляется от жестко привязанного к Земле гринвичского меридиана. Эти координаты различаются на угол Θ, представляющий собой поворот Земли за время t:

$$\alpha = \Theta_0(t) + (-\lambda) . \tag{7.12}$$

Отрицательный знак λ соответствует тому обстоятельству, что в географии долготу λ считают положительной от гринвичского меридиана к западу[1], в то время как прямое восхождение α считается положительным к востоку. Угол Θ_0 является текущим прямым восхождением нулевого меридиана, т е. звездным временем Гринвича. Вычисление звездного времени было описано в разд.3.3, где приведена соответствующая подпрограмма (LMST). Существенную трудность при этом представляет различие между всемирным временем (UT) и эфемеридным временем (ET или TDB/TDT). В то время как координаты Солнца и Луны вычисляют всегда как функцию равномерного эфемеридного времени, для вычисления звездного времени необходимо в качестве аргумента применять всемирное время. Всемирное время не может быть измерено при помощи часов, но, согласно его определению, определяется из наблюдений. Вследствие этого различие ΔT = ET − UT двух временных шкал можно точно учесть только задним числом. Для периода 1900—1985 гг. в табл 3.1 даны некоторые значения этой разности. Чтобы каждый раз не задавать значения, можно прибегнуть к аппроксимации табличных данных полиномом:

$$\Delta T = \mathrm{ET} - \mathrm{UT} = \tag{7.13}$$
$$= +71{,}28^s + 92{,}23^s\,T - 160{,}22^s\,T^2 - 516{,}52^s\,T^3 - 339{,}84^s\,T^4,$$

где T = (JD − 2451545,0)/36525,0 и $-1{,}00 \le T \le -0{,}15$.

Это приближение, имеющее точность 1—2 с, можно применять только в интервале 1900—1985 гг. Вне этого промежутка оно дает неправильные результаты. Программа ETMINUT проверяет это условие, и если оно не соблюдено, присваивает переменной VALID значение FALSE. Так как значение ET − UT нужно определять из наблюдений, предсказания на большом промежутке времени, к сожалению, невозможны. Текущие данные публикуются в разных ежегодниках, где и можно их найти.

```
(*------------------------------------------------------------------*)
(* ETMINUT: Разность эфемеридного и всемирного времен                *)
(*          (аппроксимация полиномом для 1900—1985 гг.)              *)
(*          T: время в юл. столетиях с J2000                         *)
(*          (=(JD-2451545.0)/36525.0)                                *)
(*   DTSEC: DT=ET-UT в секундах (только для VALID=TRUE)              *)
(*   VALID: TRUE для периода между 1900 и 1985 гг., иначе FALSE      *)
(*------------------------------------------------------------------*)
PROCEDURE ETMINUT(T: REAL; VAR DTSEC: REAL; VAR VALID: BOOLEAN);
  BEGIN
    VALID := ( (-1.0<T) AND (T<-0.15) );
    IF (VALID) THEN
      BEGIN
```

[1] Вариант, принятый в некоторых западных странах.—*Прим. ред.*

```
    DTSEC:= ((((-339.84*T-516.52)*T-160.22)*T)+92.23)*T+71.28;
  END;
END;
(*-----------------------------------------------------------------*)
```

Разность ET – UT меняется очень медленно и потому в течение затмения Солнца может считаться постоянной.

7.4. Продолжительность затмения

Продолжительность полного или кольцеобразного затмения не одинакова для разных точек центральной линии. Помимо диаметра тени она в большой степени зависит от скорости перемещения тени по поверхности Земли. Точное вычисление продолжительности полной фазы включает определение моментов ее наступления и окончания методом итераций и поэтому очень трудоемко. Однако для оценки хода затмения и выбора благоприятного места наблюдений на центральной линии достаточно следующих соображений.

Экваториальные координаты

$$\mathbf{r} = (x,\ y,\ z) \qquad \text{и} \qquad \mathbf{r}' = (x',\ y',\ z')$$

точек, в которых ось тени пересекает поверхность Земли в близкие друг другу моменты t и $t + \Delta t$, могут быть вычислены по уравнениям (7.5) и (7.6)[1]. Разность этих векторов является мерой скорости, с которой перемещается область тени. Правда, $\mathbf{r}' - \mathbf{r}$ еще не есть искомое перемещение тени относительно поверхности Земли.

Угол w, на который поворачивается Земля вокруг своей оси за время Δt, получим, зная, что время полного оборота равно $23^h\ 56^m$:

$$w = 2\pi \frac{\Delta t}{1436^m} \text{ (радиан)} .$$

Точка поверхности Земли, которая в момент $t + \Delta t$ имеет координаты $\mathbf{r}' = (x',\ y',\ z')$, в момент t имеет координаты

$$\mathbf{r}'' = \begin{bmatrix} +x' \cos w + y' \sin w \\ -x' \sin w + y' \cos w \\ +z' \end{bmatrix} \approx \begin{bmatrix} x' + wy' \\ y' - wx' \\ z' \end{bmatrix},$$

которые получаются из $\mathbf{r}'$ соответствующим поворотом вокруг оси z на угол w. Путь, пройденный центром тени по поверхности Земли за время Δt, будет равен

---

[1] Из-за вводимых далее дополнительных упрощений сплюснутость Земли в этом разделе не учитывается.

$$\Delta\mathbf{r} = \mathbf{r}'' - \mathbf{r}$$

или

$$\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} x' + wy' - x \\ y' - wx' - y \\ z' \quad\quad -z \end{bmatrix} .$$

Если вектор $\mathbf{e} = (e_x, e_y, e_z)$, как и в (7.3), описывает направление оси тени, $\Delta\mathbf{r}$ можно разложить на две части, одна из которых, $\Delta\mathbf{r}_{\parallel}$, параллельна $\mathbf{e}$, а другая, $\Delta\mathbf{r}_{\perp}$, перпендикулярна оси тени. Длина этих частей будет равна

$$\Delta r_{\parallel} = \Delta\mathbf{r}\cdot\mathbf{e} = \Delta x\; e_x + \Delta y\; e_y + \Delta z\; e_z ,$$

$$\Delta r_{\perp} = \sqrt{(\Delta r)^2 - (\Delta r_{\parallel})^2}.$$

$\Delta r_{\perp}$—длина отрезка, на который за время Δt перемещается точка пересечения тени перпендикулярно направлению падения тени. При диаметре конуса тени $|d|$ продолжительность τ полной или кольцеобразной фазы составляет

$$\tau = \frac{|d|}{\Delta r_{\perp}} \Delta t.$$

7.5. Координаты Солнца и Луны

Точные координаты Солнца и Луны, конечно, являются важной предпосылкой вычисления солнечных затмений. Например, чтобы определить центральную линию с точностью 10 км, нужно с такой же точностью знать положение Луны на ее орбите. При среднем расстоянии 380 000 км это соответствует углу 5″. Столь же точно должны быть известны геоцентрические координаты Солнца. Этим требованиям очень хорошо удовлетворяют представленные выше программы SUN200 и MOON. Мы не будем рассматривать здесь использованные в них разложения возмущений О них можно справиться в гл.5 и 6.

Для наших целей помимо точности требуется и достаточная скорость счета, так как для вычисления хода затмения требуется определить большое число положений Солнца и Луны. Поэтому сначала отдельные координаты надо разложить в ряд Чебышева, включающий определенные полиномы, которые в течение короткого времени представляет орбиту Солнца и Луны с достаточной точностью Вычислить их гораздо проще, чем применять обычные сложные ряды разложений Этот метод рассматривался в гл.6 при вычислении эфемерид Луны. Там обсуждались основы разложения в ряд Чебышева и соответствующие подпрограммы (T_EVAL, T_FIT_LBR и T_FIT_MOON), с которыми читатель уже освоился.

Разложение в ряд координат Луны требует лишь вызова программы T_FIT_MOON. Предполагается, что предварительно объявлен тип данных TPOLYNOM Так как затмение Солнца продолжается всегда не более нескольких часов, достаточно при разложении в ряд использовать полиномы восьмой степени.

```
CONST MAX_TP_DEG = 8;          (* порядок Т-полинома        *)
TYPE TPOLYNOM = RECORD         (* полином Чебышева          *)
           M   : INTEGER;      (* град.                     *)
           A,B : REAL;         (* интервал                  *)
             C: ARRAY [0..MAX_TP_DEG] OF REAL; (* коэффициенты*)
             END;
VAR  T_BEGIN,T_END              : REAL;
     RAM_POLY,DEM_POLY,RM_POLY:TPOLYNOM;
...

(* разложение в ряд Чебышева прямого восхождения, склонения и    *)
(* расстояния Луны ( порядок полинома MAX_TP_DEG=8)              *)
T_FIT_MOON (T_BEGIN,T_END, MAX_TP_DEG, RAM_POLY,DEM_POLY,
           RM_POLY);
...
```

Вычислить полиномы для координат Луны можно позднее с помощью подпрограммы T_EVAL.

Для такого же представления солнечных координат нам нужны еще две небольшие подпрограммы. SUNEQU вычисляет экваториальные координаты—прямое восхождение и склонение в данный момент времени. Здесь нужно отметить две поправки, необходимые для сравнения положений Солнца и Луны. Одной поправкой учитывается время распространения света от Солнца до Луны, составляющее 8,32 мин. За это время Солнце по орбите переместится примерно на 20″. Так как *наблюдаемое* положение Солнца в момент t соответствует геометрическому положению в более ранний момент $t-8{,}32^{\mathrm{m}}$ испускания света, нужно соответственно подправить и значения солнечных координат. Для Луны этот эффект уже учтен в теории движения Луны Брауна, и мы не будем подробно останавливаться на нем. После преобразования эклиптических солнечных координат в экваториальные нужно учесть и нутацию (разд.5.4.3). Исправленные таким образом прямое восхождение и склонение дают положение Солнца в системе координат, соответствующей истинному (а не среднему) положению земной оси и экватора. В подпрограмме MOONEQU таким же методом учитывается нутация для орбиты Луны.

```
(*-----------------------------------------------------------*)
(* SUNEQU: Экваториальные координаты Солнца                  *)
(*      (прямое восхождение RA, склонение DEC (град.), R(a.e.) *)
(*      T в юл. столетиях с J2000 (T:= (JD - 2451545.0)/36525) *)
(*      Координаты отнесены к истинному равноденствию даты     *)
(*-----------------------------------------------------------*)
PROCEDURE SUNEQU(T:REAL;VAR RA,DEC,R:REAL);
  VAR DT,L,B,X,Y,Z: REAL;
  BEGIN
   DT:= (8.32/1440.0)/36525.0;   (* поправка назад на 8.32 мин   *)
```

```
    SUN200(T-DT,L,B,R);          (* геоцентрические эклиптические      *)
    CART(R,B,L,X,Y,Z);           (* координаты Солнца                  *)
    ECLEQU(T,X,Y,Z);             (* экваториальные координаты          *)
    NUTEQU(T,X,Y,Z);             (* учет нутации                       *)
    POLAR(X,Y,Z,R,DEC,RA);
  END;
(*-----------------------------------------------------------------    *)
(* T_FIT_SUN: Вычисление разложения Чебышева для коорд. Солнца         *)
(*            (ряды для RA,DEC и R).                                   *)
(*                                                                     *)
(*     TA  : начало интервала разложения (юл. столет. с J2000)         *)
(*     TB  : конец интервала разложения ( TB < TA + 1 год )            *)
(*     N   : порядок полинома                                          *)
(*     RA_POLY,DE_POLY,R_POLY: полиномы Чебышева для RA,DEC,R          *)
(*-----------------------------------------------------------------    *)
PROCEDURE T_FIT_SUN ( TA,TB: REAL; N: INTEGER;
                      VAR RA_POLY,DE_POLY,R_POLY: TPOLYNOM);
  BEGIN
    T_FIT_LBR (SUNEQU,TA,TB,N,RA_POLY,DE_POLY,R_POLY);
  END;
(*-----------------------------------------------------------------    *)
```

T_FIT_SUN, подобно T_FIT_MOON, используется для вычисления разложения в ряд Чебышева солнечных координат. Из-за медленного движения Солнца достаточно применения рядов с низкими степенями.

7.6. Программа ECLIPSE

Исходя из даты ближайшего новолуния, программа ECLIPSE вычисляет среднюю линию солнечного затмения и продолжительность полной или кольцеобразной фазы.

Ядром программы является подпрограмма INTSECT, в которой по геоцентрическим координатам Солнца и Луны определяется точка пересечения оси тени с поверхностью Земли. Затем вычисляются диаметры конусов тени и полутени вблизи Земли, по которым определяют соответствующие фазы затмения. Подпрограмма CENTRAL переводит выданные INTSECT экваториальные координаты тени в географические и определяет продолжительность полной фазы затмения.

При вводе программа кроме даты новолуния запрашивает разность всемирного и эфемеридного времен и шаг выдачи. Хотя значение $\Delta T = ET - UT$ известно для прошедших моментов времени, для будущих моментов определить его трудно. Обычно при предсказаниях затмений предполагают, что $ET - UT = 0$. Вычисленные таким образом координаты точки центральной линии по географической долготе отличаются от истинных. Эта разность соответствует повороту Земли вследствие суточного вращения в течение короткого промежутка времени ΔT. Если

λ' — вычисленная долгота, то действительное ее значение при принятом здесь способе исчисления ($\lambda > 0$ к западу) будет равно

$$\lambda = \lambda' - 0{,}25^\circ/{}^m \cdot \Delta T$$

На сегодняшний день $\Delta T \approx 1^m$, и предсказанные в предположении $\Delta T = 0$ координаты центральной линии будут западнее действительных на $1/4^\circ$. На экваторе это угловое расстояние соответствует 28 км.

```
(*-------------------------------------------------------------------------*)
(*                             ECLIPSE                                      *)
(*                    Вычисление затмений Солнца                            *)
(*                        Версия 03.12.1988                                 *)
(*-------------------------------------------------------------------------*)
PROGRAM ECLIPSE(INPUT,OUTPUT);

  CONST MAX_TP_DEG= 8;          (* порядок Т-полинома                       *)
        H           = 1.14E-6;  (* 1h в юл. столет.(1/(24*36525))           *)
 TYPE REAL 33 = ARRAY [1..3, 1..3] OF REAL;
 TROLYNOM=RECORD (* полином Чебышева   *)
      M  :  INTEGER; (* степень полинома    *)
      A,B: REAL;     (* границы интервала   *)
      C  :ARRAY [0..MAX_TP_DEG] OF REAL; (* коэффициенты *)
      END;
        PHASE_TYPE= ( NO_ECLIPSE, PARTIAL,
                      NON_CEN_ANN, NON_CEN_TOT, ANNULAR, TOTAL );

  VAR T_BEGIN,T_END,T,DT,MJDUT:  REAL;
      ETDIFUT                    : REAL;
      LAMBDA,PHI,T_UMBRA          :REAL;
      RAM_POLY,DEM_POLY,RM_POLY:TPOLYNOM;
      RAS_POLY,DES_POLY,RS_POLY:TPOLYNOM;
         PHASE                    : PHASE_TYPE;
(*-------------------------------------------------------------------------*)
(* Сюда вводятся следующие подпрограммы в указанном порядке:               *)
(*   SN, CS, ASN, ATN, ATN2, CART, POLAR, GMS, T_EVAL, T_FIT_LBR*)
(*   ECLEQU                                                                 *)
(*   PMATEQU, PRECART, NUTEQU                                               *)
(*   MJD, CALDAT, LMST, ETMINUT                                             *)
(*   SUN200, SUNEQU, T_FIT_SUN                                              *)
(*   MOON, MOONEQU, T_FIT_MOON                                              *)
(*-------------------------------------------------------------------------*)

(*-------------------------------------------------------------------------*)
(* GET_INPUT: задается интервал поиска, шаг и ET—UT                        *)
(*   (T_BEGIN, T_END, DT в юл. столет. с J2000 UT; ET—UT (c)               *)
(*-------------------------------------------------------------------------*)
```

```
PROCEDURE GET_INPUT ( VAR T_BEGIN, T_END, DT, ETDIFUT: REAL );

  VAR D,M,Y      : INTEGER;
      UT,T,DTAB: REAL;
      VALID      : BOOLEAN;

  BEGIN

  WRITELN;
  WRITELN('     ECLIPSE: Вычисление затмений Солнца              ');
  WRITELN('                    Версия 03.12.88                   ');
  WRITELN('     (c) 1988 Thomas Pfleger, Oliver Montenbruck      ');
  WRITELN;
  WRITE   (' Дата новолуния (DD MM YYYY UT): '); READLN(D,M,Y,UT);
    WRITE(' шаг выдачи (мин)    : '); READLN(DTAB);
    DT      := (DTAB/1440.0)/36525.0;
    UT      := TRUNC ( UT*60.0/DTAB
               + 0 5 ) * DTAB/60.0 ; (* округление до 1 мин         *)
    T       := (MJD(D,M,Y,UT)-51544.5)/36525.0;
    T_BEGIN:= T-0.25/36525.0;     T_END:= T+0.25/36525.0;
    ETMINUT ( T, ETDIFUT, VALID);
    WRITE   (' разность ET-UT (c): ');
 IF (VALID) THEN WRITE('(предложение:',TRUNC(ETDIFUT+0.5):4,'   c)');
     READLN(ETDIFUT);
     WRITELN;
     WRITELN('   Дата      UT       phi      lambda    продолж.  фаза   ');
     WRITELN;
     WRITELN ('            h  m     o  ''     o  ''       мин           ');

  END;

(*-------------------------------------------------------------------*)
(* WRTOUT: форматированный вывод                                     *)
(*-------------------------------------------------------------------*)

PROCEDURE WRTOUT ( MJDUT,LAMBDA,PHI,T_UMBRA: REAL;
                   PHASE: PHASE_TYPE );

  VAR DAY,MONTH,YEAR,H,M:INTEGER;
      HOUR,S            : REAL;

  BEGIN
    CALDAT ( MJDUT,DAY,MONTH,YEAR,HOUR );     (* дата                *)
    WRITE  ( DAY:5,MONTH:3,YEAR:5);
    GMS(HOUR+0.5/60 0,H,M,S)
     WRITE (H:4,M:3); (* округление времени до 1 мин                 *)
    IF ( ORD(PHASE)<ORD(ANNULAR) )
```

```
      THEN
        WRITE('   -- --      -- --      ---')
      ELSE
        BEGIN
          GMS(PHI,H,M,S);   WRITE (H:9,M:3);
          GMS(LAMBDA,H,M,S); WRITE (H:7,M:3); WRITE (T_UMBRA:7:1);
        END;
    CASE PHASE OF
      NO_ECLIPSE:  WRITE(' ----                                      ');
      PARTIAL    : WRITE(' частное                                   ');
      NON_CEN_ANN: WRITE(' кольцеобразное (нецентральное)            ');
      NON_CEN_TOT: WRITE(' полное (нецентральное)                    ');
      ANNULAR    : WRITE(' кольцеобразное                            ');
      TOTAL      : WRITE(' полное                                    ');
    END;
    WRITELN;
  END;
(*-----------------------------------------------------------------------*)
(* INTSECT  Вычисление точки пересечения оси тени с поверхностью        *)
(*          Земли                                                        *)
(*                                                                       *)
(*  RAM,DEM,RM, экваториальные координаты Луны и Солнца (прямое          *)
(*  RAS,DES,RS  восх и склон (град ); расст. в радиусах Земли)           *)
(*  X,Y,Z       экватор.коорд. точки тени (в радиусах Земли)             *)
(*  EX,EY,EZ  : вектор направления оси тени                              *)
(*  D_UMBRA   : диаметр тени в радиусах Земли                            *)
(*  PHASE     : фаза затмения                                            *)
(*-----------------------------------------------------------------------*)

PROCEDURE INTSECT ( RAM,DEM,RM,RAS,DES,RS         : REAL;
                    VAR X,Y,Z,    EX,EY,EZ, D_UMBRA:REAL;
                    VAR PHASE:  PHASE_TYPE );

  CONST FAC= 0.996633;  (* отношение полярного радиуса Земли к    *)
                        (* экваториальному                        *)
        D_M=   0.5450;  (* диаметр Луны в радиусах Земли          *)
        D_S = 218.25;   (* диаметр Солнца в радиусах Земли        *)

  VAR XM,YM,ZM, XS,YS,ZS, XMS,YMS,ZMS, RMS:REAL;
      DELTA, R0, S0, S, D_PENUMBRA      : REAL;

  BEGIN

    CART(RM,DEM,RAM,XM,YM,ZM);
            ZM:=ZM/FAC;            (* координаты Солнца и Луны;  *)
    CART(RS,DES,RAS,XS,YS,ZS);
            ZS:=ZS/FAC;            (* шкала z-оси                *)
```

```
    XMS:=XM-XS;  YMS:=YM-YS;  ZMS:=ZM-ZS;   (* вектор,                   *)
    RMS:=  SQRT(XMS*XMS+YMS*YMS+ZMS*ZMS);  (* расстояние и              *)
    EX  :=XMS/RMS;   EY:=YMS/RMS;
                EZ:=ZMS/RMS;       (* вектор направления Солнце ->Луна  *)

  S0     := -( XM*EX + YM*EY + ZM*EZ )   (* расст.Луна—фунд. пл.        *)
  DELTA:= S0*S0+1.0-XM*XM-YM*YM-ZM*ZM;
  R0     := SQRT(1.0-DELTA);          (* расст. центр Земли—ось тени    *)

  D_UMBRA := (D_S-D_M)*S0/RMS-D_M;(* диаметры тени и полутени *)
  D_PENUMBRA:=(D_S+D_M)*S0/RMS+D_M; (* на фундамент. плоскости *)

(* определение фазы и при надобности вычисление координат тени      *)
    IF ( R0 < 1.0 )
      THEN                       (* ось пересекает поверхность Земли   *)
      BEGIN                  (* ->полное или кольцеобразное затмение   *)
      S          := S0-SQRT(DELTA);
      D_UMBRA:= (D_S-D_M)*(S/RMS)-D_M; (* диаметр тени на Земле *)
          X   :=XM+S*EX;   Y:=YM+S*EY;   Z:=ZM+S*EZ;
          Z   :=Z*FAC;                    (* изменение шкалы z-оси  *)
          IF D_UMBRA>0 THEN PHASE:=ANNULAR ELSE PHASE:=TOTAL;
        END
      ELSE
        IF ( R0 < 1.0+0.5*ABS(D_UMBRA) )
           THEN                 (* нецентральное затмение              *)
IF D_UMBRA>0 THEN PHASE:=NON_CEN_ANN ELSE PHASE:=NON_CEN_TOT
          ELSE
             IF ( R0 < 1.0+0.5*D_PENUMBRA)
     THEN PHASE:=  PARTIAL      (* частное затмение                    *)
      ELSE PHASE:=  NO_ECLIPSE; (* затмения нет                        *)

  END;

(*----------------------------------------------------------------------*)
(* CENTRAL: центральная линия, фаза и продолжительность затмения       *)
(*                                                                      *)
(* T_UT    : время в юлианских столетиях с J2000 UT                     *)
(* ETDIFUT: разность эфемеридное (ET)—всемирное (UT) время (с)          *)
(* RAM_POLY,DEM_POLY,RM_POLY,  RAS_POLY,DES_POLY,RS_POLY:               *)
(*              полиномы Чебышева для координат Солнца и Луны           *)
(* LAMBDA, PHI: географ. долгота и широта центра тени (град.)           *)
(* T_UMBRA: продолжит. полного/кольцеобр. затмения (мин)                *)
(* PHASE  : фаза затмения                                               *)
(*----------------------------------------------------------------------*)

PROCEDURE CENTRAL ( T_UT, ETDIFUT          : REAL;
                         RAM_POLY,DEM_POLY,RM_POLY:TPOLYNOM;
```

```
                  RAS_POLY,DES_POLY,RS_POLY:TPOLYNOM;
                  VAR LAMBDA,PHI,T_UMBRA:  REAL;
                  VAR PHASE               : PHASE_TYPE );

CONST AE = 23454.78;     (* 1 а.е. в радиусах Земли                *)
                         (* (149597870км/6378.14км)                *)
    DT   = 0.1;          (* малый интервал времени; dt = 0.1 мин   *)
    MPC = 52596000.0;    (* мин. в юл. столетии (1440*36525)       *)
    OMEGA=4.3755E-3;     (* угловая скорость Земли (рад./мин)      *)

  VAR RAM,DEM,RM, RAS,DES,RS, RA,DEC,R, DX,DY,DZ,D, MJDUT:  REAL;
      T,X,Y,Z,EX,EY,EZ,D_UMBRA,  XX,YY,ZZ,EXX,EYY,EZZ,DU,  W: REAL;
      PH: PHASE_TYPE;

 (* вычисление координат Солнца и Луны, через полиномы Чебышева *)
 PROCEDURE POSITION ( T:REAL; VAR RAM,DEM,RM, RAS,DES,RS : REAL );
    BEGIN
      RAM:= T_EVAL(RAM_POLY,T); RAS:= T_EVAL(RAS_POLY,T);
      DEM:= T_EVAL(DEM_POLY,T); DES:= T_EVAL(DES_POLY,T);
      RM:=  T_EVAL(RM_POLY,T);  RS :=  T_EVAL(RS_POLY,T);  RS:=RS*AE;
    END;

  BEGIN

    (* юлианские столетия с J2000 ET      *)
    T:=  T_UT  +  ETDIFUT/(86400.0*36525.0);

    (* фаза затмения и координаты тени в момент T        *)
    POSITION ( T, RAM,DEM,RM, RAS,DES,RS );
 INTSECT ( RAM,DEM,RM, RAS,DES,RS, X,Y,Z,EX,EY,EZ, D_UMBRA,PHASE );

(* только для центральной фазы: географические координаты и        *)
(*  продолжительность полной фазы                                   *)
    IF ( ORD(PHASE) < ORD(ANNULAR) )
       THEN BEGIN LAMBDA:=0.0; PHI:=0.0; T_UMBRA:=0.0; END
       ELSE
         BEGIN
            (* географические координаты:                            *)
            MJDUT:=  36525.0*T_UT  +  51544.5;
            POLAR (  X,Y,Z,  R,DEC,RA  );
            PHI    =  DEC  +  0.1924*SN(2.0*DEC);
            LAMBDA:=15.0*LMST(MJDUT,0.0)-RA;
            IF LAMBDA>+180.0 THEN LAMBDA:=LAMBDA-360.0;
            IF LAMBDA<-180.0 THEN LAMBDA:=LAMBDA+360.0;
(* продолжительность полной фазы для истинного положения тени:     *)
            (* (a) координаты тени для момента T+DT (или T-DT)       *)
   POSITION (  T+DT/MPC,  RAM,DEM,RM,  RAS,DES,RS  ); W:=+DT*OMEGA;
```

```
  INTSECT    (RAM,DEM,RM, RAS,DES,RS, XX, YY,ZZ,EXX,EYY,EZZ,DU,PH);
          IF (ORD(PH)<ORD(ANNULAR)) THEN
             BEGIN
  POSITION  (T-DT/MPC, RAM,DEM,RM,RAS,DES,RS); W:=-DT*OMEGA;
 INTSECT    ( RAM,DEM,RM,RAS,DES,RS, XX,YY,ZZ,EXX,EYY,EZZ, DU,PH );
             END;
  (* (b)  сдвиг DX,DY,DZ тени на Земле                          *)
 (*       и часть D, перпендикулярная оси тени                  *)
    DX:= XX-X+W*Y;    DY:= YY-Y-W*X;     DZ:= ZZ-Z;
    D := SQRT( DX*DX+DY*DY+DZ*DZ -
    (DX*EX+DY*EY+DZ*EZ)*(DX*EX+DY*EY+DZ*EZ) );
    T_UMBRA:= DT * ABS(D_UMBRA) / D;
       END
  END;

(*--------------------------------------------------------------*)

BEGIN (* основная программа       *)

  (* считывание интервала поиска   *)
  GET_INPUT ( T_BEGIN,T_END, DT, ETDIFUT );

  (* разложение в ряд Чебышева    *)

  T_FIT_MOON(T_BEGIN-H,T_END+H,8,RAM_POLY,DEM_POLY,RM_POLY);
  T_FIT_SUN  (T_BEGIN-H,T_END+H,3,RAS_POLY,DES_POLY,RS_POLY);


  (* вычисление фазы и центральной линии затмения               *)
  T:= T_BEGIN,

  REPEAT

    CENTRAL( T,ETDIFUT,
             RAM_POLY,DEM_POLY,RM_POLY, RAS_POLY,DES_POLY,RS_PO
             LAMBDA, PHI, T_UMBRA, PHASE);

    IF PHASE<>NO_ECLIPSE THEN
      BEGIN
        MJDUT:= 36525.0*T  +  51544.5;
        WRTOUT( MJDUT, LAMBDA, PHI, T_UMBRA, PHASE );
      END;
    T:= T+DT;
  UNTIL (T > T_END);
END.
(*--------------------------------------------------------------*)
```

В качестве примера вычислим ход полного затмения Солнца, которое можно будет наблюдать в 1999 г на юге Германии. Центральная линия этого затмения будет проходить вблизи городов Штутгарт и Мюнхен Для определения дат новолуний в 1999 г. вызовем сначала NEWMOON. Зная эклиптическую широту Луны, можно судить о сроках возможных затмений. Эмпирическое правило состоит в том, что полное или кольцеобразное затмение возможно при широте Луны менее одного градуса. Частное затмение может быть при широте не более 1,5°.

NEWMOON. Моменты новолуния и эклиптическая широта Луны
Версия 12.11.88
(c) 1988 Thomas Pfleger, Oliver Montenbruck

Моменты новолуния для года *1999*

Дата			UT	Широта
			h	°
17	1	1999	15.8	2.2
16	2	1999	6 7	-0.5
17	3	1999	18 7	-3.0
16	4	1999	4 2	-4.6
15	5	1999	12.0	5.0
13	6	1999	19 2	-4 0
13	7	1999	2 6	-2 1
11	8	1999	11 2	0 4
9	9	1999	22.0	2 9
9	10	1999	11 7	4 6
8	11	1999	4.0	5 0
7	12	1999	22.6	3 9

16 февраля и 11 августа 1999 г. эклиптическая широта Луны при новолунии так мала, что затмения должны произойти Детальное исследование при помощи программы ECLIPSE позволяет определить области, где эти затмения будут наблюдаться

16 02.1999. Юг Индийского Океана, острова Принца Эдуарда, острова Крозе, Австралия (кольцеобразное)

11.08.1999 Север Атлантики, юг Великобритании, Центральная Европа, Ближний Восток, Индия (полное затмение)

Теперь нужно определить с помощью программы ECLIPSE ход полного затмения 11 августа 1999 г Для этого нужно задать уже вычисленную дату новолуния, желаемый шаг в минутах и значение разности между эфемеридным и всемирным временем. Аппроксимация ΔT = ET − UT, вычисляемая подпрограммой ETMINUT, может дать хорошие результаты только в интервале 1900—1985 гг. Для исследуемого затмения зададим значение 60^s. Эта величина и все остальные вводные данные приводятся курсивом. В дальнейшем некоторые значения времени частной фазы в целях экономии места опущены.

ECLIPSE: Вычисление затмений Солнца
Версия 03.12.88
(c) 1988 Thomas Pfleger, Oliver Montenbruck

дата новолуния (DD MM YYYY UT): *11 8 1999 11.2*
шаг выдачи (мин) : *3.0*
разность ET - UT (c) : *60.0*

Дата			UT		phi		lambda		продолж.	фаза
			h	m	°	'	°	'	мин	
11	8	1999	8	27	--	--	--	--	---	частное
11	8	1999	8	30	--	--	--	--	---	частное
.		.		.	.		.		.	.
.	.			.	.		.		.	.
.		.		.	.		.		.	.
11	8	1999	9	24	--	--	--	--	---	частное
11	8	1999	9	27	--	--	--	--	---	частное
11	8	1999	9	30	--	--	--	--	---	полное **(нецентральное)**
11	8	1999	9	33	44	51	50	36	1.1	полное
11	8	1999	9	36	46	23	43	47	1.2	полное
11	8	1999	9	39	47	24	38	36	1.4	полное
11	8	1999	9	42	48	9	34	14	1.5	полное
11	8	1999	9	45	48	44	30	22	1.5	полное
11	8	1999	9	48	49	11	26	52	1.6	полное
11	8	1999	9	51	49	32	23	37	1.7	полное
11	8	1999	9	54	49	47	20	36	1.8	полное
11	8	1999	9	57	49	59	17	44	1.8	полное
11	8	1999	10	0	50	7	15	2	1.9	полное
11	8	1999	10	3	50	11	12	27	1.9	полное
11	8	1999	10	6	50	13	9	59	2.0	полное
11	8	1999	10	9	50	12	7	38	2.1	полное
11	8	1999	10	12	50	9	5	21	2.1	полное
11	8	1999	10	15	50	3	3	10	2.1	полное
11	8	1999	10	18	49	56	1	3	2.2	полное
11	8	1999	10	21	49	46	0	-58	2.2	полное
11	8	1999	10	24	49	35	-2	56	2.3	полное
11	8	1999	10	27	49	22	-4	51	2.3	полное
11	8	1999	10	30	49	7	-6	43	2.3	полное
11	8	1999	10	33	48	52	-8	31	2.3	полное
11	8	1999	10	36	48	34	-10	16	2.4	полное
11	8	1999	10	39	48	16	-11	58	2.4	полное
11	8	1999	10	42	47	56	-13	38	2.4	полное
11	8	1999	10	45	47	35	-15	15	2.4	полное
11	8	1999	10	48	47	12	-16	50	2.4	полное
11	8	1999	10	51	46	49	-18	22	2.4	полное
11	8	1999	10	54	46	25	-19	53	2.4	полное
11	8	1999	10	57	45	59	-21	21	2.4	полное
11	8	1999	11	0	45	33	-22	48	2.5	полное
11	8	1999	11	3	45	6	-24	14	2.4	полное

11	8	1999	11	6	44	37	-25	38	2.4	полное
11	8	1999	11	9	44	8	-27	0	2.4	полное
11	8	1999	11	12	43	38	-28	22	2.4	полное
11	8	1999	11	15	43	7	-29	42	2.4	полное
11	8	1999	11	18	42	35	-31	1	2 4	полное
11	8	1999	11	21	42	2	-32	20	2.4	полное
11	8	1999	11	24	41	29	-33	38	2.4	полное
11	8	1999	11	27	40	54	-34	56	2.3	полное
11	8	1999	11	30	40	19	-36	13	2.3	полное
11	8	1999	11	33	39	42	-37	30	2.3	полное
11	8	1999	11	36	39	5	-38	48	2.3	полное
11	8	1999	11	39	38	27	-40	5	2.2	полное
11	8	1999	11	42	37	47	-41	23	2 2	полное
11	8	1999	11	45	37	7	-42	42	2 1	полное
11	8	1999	11	48	36	26	-44	2	2.1	полное
11	8	1999	11	51	35	43	-45	23	2.1	полное
11	8	1999	11	54	34	59	-46	46	2.0	полное
11	8	1999	11	57	34	14	-48	10	2.0	полное
11	8	1999	12	0	33	28	-49	38	1 9	полное
11	8	1999	12	3	32	40	-51	8	1.9	полное
11	8	1999	12	6	31	50	-52	41	1.8	полное
11	8	1999	12	9	30	58	-54	20	1.7	полное
11	8	1999	12	12	30	4	-56	3	1.7	полное
11	8	1999	12	15	29	8	-57	54	1.6	полное
11	8	1999	12	18	28	8	-59	53	1.5	полное
11	8	1999	12	21	27	4	-62	4	1.5	полное
11	8	1999	12	24	25	55	-64	31	1.4	полное
11	8	1999	12	27	24	39	-67	19	1.3	полное
11	8	1999	12	30	23	12	-70	44	1.2	полное
11	8	1999	12	33	21	24	-75	20	1.0	полное
11	8	1999	12	36	--	--	--	--	---	полное **(нецентральное)**
11	8	1999	12	39	--	--	--	--	---	частное
11	8	1999	12	42	--	--	--	--	---	частное
.	.	.	.	.	.		.		.	.
.	.	.	.	.	.		.		.	.
.	.	.	.	.	.		.		.	.
11	8	1999	13	36	--	--	--	--	---	частное
11	8	1999	13	39	--	--	--	--	---	частное

Если фаза затмения полная, но не центральная, то ось конуса тени проходит мимо Земли, но на ее поверхности имеется область, которая попадает в тень. Обычно это случается в начале или в конце затмения. Затмения, которые наблюдаются как полные или кольцеобразные, но не центральные, так как конус тени только касается Земли, очень редки. Такие затмения очень трудны для наблюдений, так как они непродолжительны и происходят на высоких широтах.

8. ПОКРЫТИЯ ЗВЕЗД

В течение суток Луна перемещается по орбите с запада на восток примерно на 13°. Проследить это движение легче всего, если Луна перемещается мимо ярких звезд. Перемещение особенно заметно при покрытиях звезд, когда они скрываются за восточным краем Луны и через некоторое время появляются с другого ее края. В общей сложности более тысячи звезд, видимых невооруженным глазом, могут быть покрыты Луной Все они лежат в узкой полосе не далее 8° к северу и югу от эклиптики.

Покрытия звезд во многом похожи на затмения Солнца. Однако в параллельных лучах далекой звезды тень Луны не сходится в конус, а имеет везде одинаковый диаметр. По той же причине нет разницы между тенью и полутенью. Тень, отбрасываемая Луной в свете звезды, такого же размера, что и сама Луна (примерно 1/4 диаметра Земли), и потому не может покрыть всю Землю. Следовательно, покрытие определенной звезды можно наблюдать лишь с части земной поверхности.

Поскольку у Луны нет атмосферы, звезды покрываются не постепенно, а сразу и так же сразу появляются вновь. Время покрытия и появления легко измерить, что позволяет уточнить положение Луны. Наблюдения покрытий звезд ведутся уже давно, обогащая наши знания о движении Луны и вращении Земли.

При планировании наблюдений можно воспользоваться данными, опубликованными в астрономических ежегодниках и журналах. Правда, в них приводятся сведения лишь для некоторых мест, в большинстве случаев для больших городов С помощью так называемого коэффициента места наблюдения можно пересчитать время начала и конца покрытия для любой местности. Вычисления подобных предсказаний обычно не требуют большой точности, поэтому, вместо того чтобы следовать строгой процедуре вычислений, допускают некоторые упрощения. Например, можно не учитывать неровность края Луны Тем не менее предсказание покрытий является довольно трудоемкой задачей, так как необходимо в заданном промежутке времени проверить много звезд на предмет покрытия их Луной.

Программа OCCULT позволяет вычислять возможные покрытия любого числа звезд в интервале десяти суток. В этом временно́м интервале, равном одной трети времени обращения Луны, возможно не более одной встречи Луны с определенной звездой. Наименьшее расстояние между Луной и звездой достигается в момент, когда геоцентрическое прямое восхождение Луны равно прямому восхождению звезды. Этот момент может быть определен простой итерацией. Если при этом разность склонений не очень велика, то можно предположить, что покрытие будет видно хотя бы на части земной поверхности. Наконец, можно исследовать движение тени Луны вблизи момента соединения и установить, наступит ли покрытие в заданном месте наблюдения.

8.1. Видимые координаты звезд

Координаты звезды, для которой вычисляется момент покрытия, можно найти в ряде каталогов. Примером тому служат SAO—каталог Смитсонианской астрофизической обсерватории и Зодиакальный каталог (Zodiacal Catalogue—каталог 3539 зодиакальных звезд для равноденствия 1950.0), специально составленный для расчета моментов покрытия звезд. Небольшая часть Зодиакального каталога с основными данными о звездах Плеяд воспроизведена в табл.8.1. Наряду с прямым восхождением и склонением для равноденствия 1950.0 в таблице приведены собственные движения звезд за сто лет. Как видно из таблицы, пространственное движение звезд относительно Солнечной системы со временем приводит к заметным изменениям их координат. Поэтому сначала необходимо, выяснив собственное движение, пересчитать положение звезды на приблизительное время покрытия.

Таблица 8.1 Координаты и собственные движения (PM) за 100 лет некоторых звезд Плеяд (по Зодиакальному каталогу)

№	Прямое восхож. (1950)			PM	Склонение (1950)			PM	Примечания
	h	m	s	s	°	′	″	″	
536	3	41	49,540	+0,110	+24	8	1,58	-4,26	Целено
537	3	41	54,063	+0,144	+23	57	29,90	-4,16	Электра
539	3	42	13,603	+0,171	+24	18	43,07	-4,20	Тайгета
541	3	42	50,749	+0,127	+24	12	46,74	-4,78	Майя
542	3	42	55,387	+0,100	+24	23	59,69	-4,91	Астеропа
545	3	43	21,195	+0,145	+23	47	38,98	-4,34	Меропа
552	3	44	30,427	+0,144	+23	57	7,52	-4,47	Альциона
560	3	46	11,022	+0,111	+23	54	7,41	-4,71	Атлас
561	3	46	12,393	+0,114	+23	59	7,18	-5,53	Плейона

Положения, приведенные в каталогах, даже после исправления нельзя непосредственно использовать для вычисления покрытий звезд. Для сравнения положений звезды и Луны нужны так называемые *видимые* координаты рассматриваемых звезд. Они исчисляются относительно текущих положений точки весеннего равноденствия и небесного экватора, определенных с учетом прецессии и нутации для равноденствия даты. При этом система координат ориентирована параллельно оси вращения Земли в данный момент. Однако в звездных каталогах положения даны в координатной системе, ориентированной по среднему экватору и точке весеннего равноденствия в определенную эпоху. Во многих случаях используются эпоха и равноденствие 1950 г., но уже существуют отдельные каталоги, в которых применяется введенное в 1984 г. равноденствие 2000.

Прецессия, выражающая среднее вековое перемещение точки весеннего равноденствия, может быть вычислена при помощи уравнений (2.10) и (2.11). Преобразование координат звезды со средним равноденствием каталога в координаты со средним равноденствием любой даты осуществляют, применяя подпрограммы PMATEQU и PRECART. Из-за переменности сил тяготения Солнца и Луны на прецессию земной оси накладываются еще и периодические колебания—нутация. Она описывается двумя углами, $\Delta\varepsilon$ и $\Delta\psi$, максимальное значение которых достигает соответственно 9″ и 17″ (5.16). Средние координаты звезд отличаются от истинных на величину именно такого порядка. Пересчет координат может быть осуществлен по формуле (5.17). Подпрограммы PMATEQU и NUTEQU можно объединить в одну общую программу PN_MATRIX, которая дает матрицу, преобразующую звездные координаты со средним равноденствием каталога в координаты с истинным равноденствием даты. Эта матрица определена как массив типа REAL33, который надо описать один раз в начале программы.

```
TYPE REAL33 = ARRAY[1..3,1..3] OF REAL;
(*-----------------------------------------------------------------*)
(* PN_MATRIX: комбинированная матрица прецессии и нутации для      *)
(*            преобразования координат от среднего                  *)
(*            равноденствия T0 к истинному равноденствию T          *)
(*     ( T0, T в юл. столетиях с J2000; T=(JD-2451545.0)/36525 )    *)
(*-----------------------------------------------------------------*)
PROCEDURE PN_MATRIX ( T0,T:REAL; VAR A: REAL33 );
  BEGIN
    PMATEQU(T0,T,A);                        (* прецессионная матрица T0->T*)
    NUTEQU(T,A[1,1],A[2,1],A[3,1]);
    NUTEQU(T,A[1,2],A[2,2],A[3,2]);         (*нутационное преобразование *)
    NUTEQU(T,A[1,3],A[2,3],A[3,3]);         (* трех вектор-столбцов A    *)
  END;
(*-----------------------------------------------------------------*)
```

Используют программу в общем так же, как и PMATEQU, что видно из следующего примера:

```
VAR RA,DEC,R,X,Y,Z;TEQX,T: REAL;
    A                    : REAL33;
...

TEQX := -0.5;                 (* равноденствие каталога 1950       *)
READ (RA,DEC);                (* прямое восхожд. и склонение (град.) *)
READ(T);                      (* дата в юл. столетиях с J2000      *)
PN_MATRIX (TEQX, T, A);       (* вычисление матрицы                *)
CART ( 1.0,DEC,RA,X,Y,Z);     (* прямоугольные координаты звезды   *)
PRECART ( A, X,Y,Z );         (* умножение на матрицу              *)
POLAR(X,Y,Z, R,DEC,RA );      (* новые полярные координаты RA и DEC*)
WRITELN (RA, DEC );
```

Следующая необходимая поправка координат связана с аберрацией, возникающей из-за конечного значения скорости света. Для наблюдателя, движущегося вместе с Землей вокруг Солнца, направление прихода света от звезды отличается от направления, которое регистрирует наблюдатель, покоящийся относительно Солнца. Наблюдаемое положение звезды определяется достаточно точно, если к вектору $\mathbf{e} = (x, y, z)$ прямоугольных звездных координат прибавить отношение $\mathbf{v}_{\oplus}/c$ скорости Земли к скорости света. После этого определяется прямое восхождение и склонение с учетом аберрации. При упрощающем предположении, что Земля движется вокруг Солнца по круговой орбите, скорость Земли (выраженную в экваториальных координатах) получают из равенств

$$\begin{aligned} v_{\oplus x}/c &= -0{,}994 \cdot 10^{-4} \sin(L), \\ v_{\oplus y}/c &= +0{,}912 \cdot 10^{-4} \cos(L), \\ v_{\oplus z}/c &= +0{,}395 \cdot 10^{-4} \cos(L), \end{aligned} \tag{8.1}$$

где

$$L = 2\pi\,(0{,}27908 + 100{,}00214T)$$

обозначает гелиоцентрическую долготу Земли (в радианах). Здесь T, как обычно, число юлианских столетий начиная с эпохи J2000.

```
(*-----------------------------------------------------------------*)
(* ABERRAT: Вектор скорости Земли (экваториальный, в единицах      *)
(*          скорости света )                                        *)
(*-----------------------------------------------------------------*)
PROCEDURE ABERRAT(T: REAL; VAR VX,VY,VZ: REAL);
  CONST P2=6.283185307;
  VAR L,CL: REAL;
  FUNCTION FRAC(X:REAL):REAL;
    BEGIN X:=X-TRUNC(X); IF (X<0) THEN X :=X+1; FRAC:=X  END;
  BEGIN
    L := P2*FRAC(0.27908+100.00214*T);     CL:=COS(L);
    VX:= -0.994E-4*SIN(L); VY:=  +0.912E-4*CL; VZ:=  +0.395E-4*CL;
  END;
(*-----------------------------------------------------------------*)
```

Полное исправление координат, взятых из звездного каталога, за прецессию, нутацию и аберрацию, можно произвести, как в следующем примере.

```
VAR RA,DEC,R,X,Y,Z,  VX,VY,VZ,  TEQX,T:  REAL;
    A                                 :  REAL 33;


TEQX := -0,5;               (* равнод. 1950 в юл. столетиях с J2000  *)
READ (RA,DEC);              (* прямое восхождение                     *)
                            (* и склонение из каталога                *)
                            (* со средним равноденствием 1950         *)
```

```
READ(T),                          (* дата в юл. столетиях с J2000          *)
PN_MATRIX (TEQX,T,A);             (* матрица для выч. прецессии и нутации  *)
ABERRAT (T,VX,VY,VZ);             (* скорость Земли                        *)
CART (1.0,DEC,RA,X,Y,Z);          (* прямоугольные координаты звезды       *)
PRECART (A,X,Y,Z);                (* умножение на матрицу прец. и нутации  *)
X =X*VX, Y=Y*VY, Z=Z+VZ;          (* аберрация                             *)
POLAR(X,Y,Z,R,DEC,RA);            (* видимые координаты RA и DEC           *)
WRITELN (RA, DEC),
```

8.2. Соединение в геоцентрических координатах

Для предсказания покрытий звезд из большого числа эклиптических звезд нужно выбрать те, которые в данном промежутке времени могут быть покрыты Луной С этой целью для каждой звезды находят момент, когда ее прямое восхождение α_* совпадает с геоцентрическим прямым восхождением α_M Луны Сравнение склонений Луны и звезды в момент такого соединения позволяет затем определить, будет ли видно покрытие звезды, по крайней мере на части земной поверхности. Кроме того, момент соединения служит исходной точкой для точного вычисления моментов возможных покрытий, которое будет рассмотрено в следующем разделе

От момента t_1 до момента t_2 прямое восхождение Луны меняется между $\alpha_M(t_1)$ и $\alpha_M(t_2)$ Если прямое восхождение звезды α_* заключено между этими двумя значениями, то момент соединения t_c определяется линейной интерполяцией (рис 8 1). Так как прямое восхождение Луны меняется очень равномерно, то можно использовать следующее приближение:

$$t_c \approx t' = t_2 - (\alpha_M(t_2) - \alpha_*) \cdot \frac{t_2 - t_1}{\alpha_M(t_2) - \alpha_M(t_1)} .$$

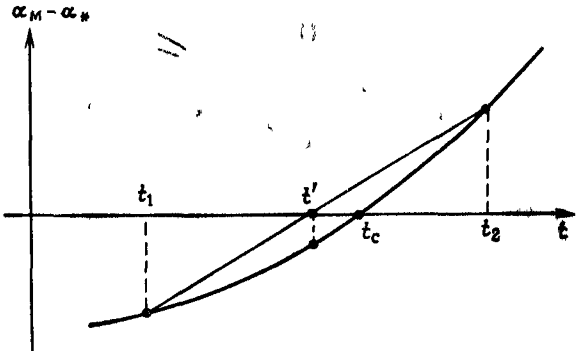


Рис. 8.1. Итеративный метод вычисления момента соединения.

Теперь, в зависимости от того что больше—прямое восхождение Луны в момент t' или же прямое восхождение звезды, — t_1 или t_2 заменяют на t'. Вновь

$$\alpha_M(t_1) \le \alpha_* \le \alpha_M(t_2),$$

и можно повторить процедуру уточнения момента покрытия. Благодаря равномерности движения Луны этот метод, называемый методом ложного положения (regula falsi), быстро и уверенно приводит к желаемому результату.

Вероятность покрытия теперь можно определить, исходя из координат Луны в момент соединения. Как видно из рис.8.2, тень Луны попадает на Землю, если склонение Луны не очень отличается от склонения звезды. При этом расстояние d оси тени от центра Земли не может быть больше, чем сумма радиуса Луны R_M и радиуса Земли $R_\oplus$:

$$d < R_\oplus + R_M \approx 1{,}3\ R_\oplus$$

В момент соединения по прямому восхождению должно соблюдаться условие

$$d(t_c) = r_M\ |\sin(\delta_M - \delta_*)| < R_\oplus + R_M \approx 1{,}3\ R_\oplus$$

Из-за наклона орбиты Луны к небесному экватору $d(t_c)$, правда, может быть на 0,2 земных радиуса больше, чем наименьшее расстояние тени Луны от центра Земли. Выбрать возможное покрытие можно, используя следующее условие

$$r_M\ |\sin(\delta_M - \delta_*)| \begin{Bmatrix} < \\ > \end{Bmatrix} 1{,}5R_\oplus \Rightarrow \begin{Bmatrix} \text{покрытие возможно} \\ \text{покрытия нет} \end{Bmatrix} \qquad (8.2)$$

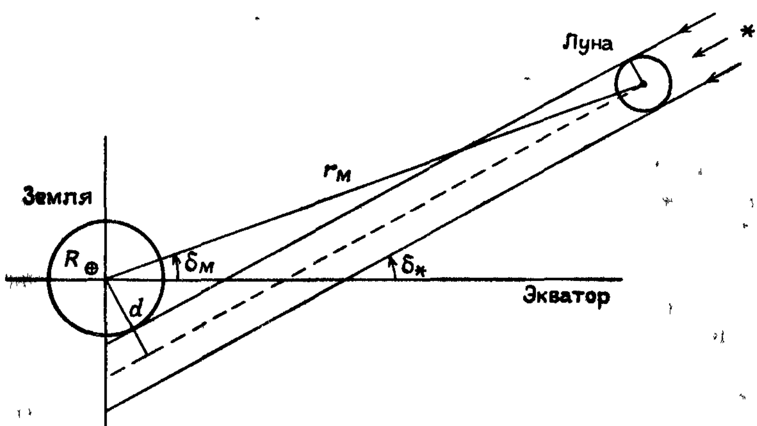


Рис.8.2. Положение Луны в момент соединения по прямому восхождению.

При поиске момента покрытия, как и на следующих стадиях вычисления, нужно определить всю серию положений Луны. Поэтому выгодно возвратиться к уже рассмотренному разложению координат Луны по полиномам Чебышева (гл.6). Тогда по немногим точно определенным положениям получается довольно простое представление лунной орбиты полиномом, который затем можно использовать для сколь угодно большого числа моментов времени.

Для этого мы уже имеем подпрограмму T_FIT_MOON, которая сама вызывает подпрограммы MOON, MOONEQU и T_FIT_LBR. Чтобы их использовать, тип данных TPOLYNOM нужно описать глобально:

```
CONST  MAX_TP_DEG = 13;
TYPE TPOLYNOM = RECORD                          (* полином Чебышева *)
            M:  INTEGER;                        (* степень          *)
          A,B:  REAL;                           (* интервал         *)
          C  :  ARRAY [0..MAX_TP_DEG] OF REAL;  (* коэффициенты     *)
 END;
```

Порядок полинома при этом выбирается таким образом, чтобы координаты Луны можно было бы вычислить в интервале десяти суток без потери точности. Следующие строки программы:

```
VAR TA,TB: REAL;
    RA_POLY,DE_POLY,
...

T_FIT_MOON (TA,TB,MAX_TP_DEG,RA_POLY,DE_POLY,R_POLY);
```

позволяют определить коэффициенты для вычисления экваториальных координат Луны — прямого восхождения, склонения и геоцентрического расстояния. Для любого момента T внутри интервала разложения, используя подпрограмму T_EVAL, можно вычислить соответствующий полином.

```
RA := T_EVAL (RA_POLY, T);    (* прямое восхождение (град.)       *)
                              (* -360 < =RA < =+360               *)
DEC:= T_EVAL (DE_POLY, T);    (* склонение в град.                *)
R  := T_EVAL (R_POLY,  T);    (* расстояние в радиусах Земли      *)
```

Определенные таким образом координаты относятся к истинному равноденствию даты, т. е. к текущему положению точки весеннего равноденствия и небесного экватора с учетом прецессии и нутации.

Теперь можно написать первую часть программы поиска покрытий звезд. Процедура CONJUNCT сначала проверяет, лежит ли прямое восхождение звезды в области прямых восхождений Луны между моментами T_1 и T_2. Если это условие выполняется, то затем ищут момент соединения и устанавливают, встречается ли тень Луны с Землей. Если все эти условия соблюдены, то момент соединения передается в вызывающую программу.

```
(*-------------------------------------------------------------  *)
(*  CONJUNCT:                                                     *)
(*  проверяет, случится ли в интервале времени [TA,TB] соединение *)
(*  Луны и звезды, при котором тень Луны падает на Землю, и       *)
(*  вычисляет момент такого соединения                            *)
(*  TA,TB:      интервал поиска                                   *)
(*  RAPOLY,DEPOLY,RPOLY: полиномы Чебышева для координат Луны     *)
(*  RA,DEC:  прямое восхождение и склонение звезды (0<=RA<=360)   *)
(*  CONJ:TRUE/FALSE (соединение есть/соединения нет)              *)
(*  T_CONJ: момент соединения по прямому восхождению              *)
(*          (=0.0, если  нет соединения)                          *)
(*  Все моменты времени - в юлианских столетиях с J2000           *)
(*  Разложение прямого восхождения должно давать значения         *)
(*  в области -360<=RA<=+360 градусов и покрывать                 *)
(*  менее одного оборота                                          *)
(*-------------------------------------------------------------  *)
PROCEDURECONJUNCT (TA,TB: REAL; RAPOLY,DEPOLY,RPOLY:TPOLYNOM;
           RA,DEC: REAL; VAR CONJ: BOOLEAN; VAR T_CONJ: REAL);
  CONST EPS=1E-3;  (* точность RA (град.)                          *)
  VAR  RA_A,RA_B                        : REAL;
       T1,T2,T_NEW,DRA1,DRA2,DRA_NEW:REAL;
       DE_CON, R_CON                    : REAL;
  BEGIN

  T_CONJ:=0.0;  RA_A=T_EVAL(RAPOLY,TA); RA_B=T_EVAL(RAPOLY,TB);

   (* проверка RA_A<= RA <= RA_B                                   *)
  CONJ := (RA_A<=RA) AND (RA<=RA_B);
  IF (NOT CONJ) THEN(* проверка еще раз для (RA - 360 град)       *)
   BEGIN RA:=RA-360.0; CONJ :=((RA_A<=RA) AND (RA<=RA_B)); END;

  IF CONJ THEN
   BEGIN
    (* определение момента соединения методом "regula falsi"      *)
    (* ( [T1,T2] всегда включает в себя T_CONJ )                  *)
  T1:=TA;  DRA1:= RA_A-RA;
  T2:=TB;  DRA2:= RA_B-RA;
  REPEAT
   T_NEW := T2 - DRA2*(T2-T1)/(DRA2-DRA1);
   DRA_NEW:=T_EVAL(RAPOLY,T_NEW) - RA;
   IF DRA_NEW>0 THEN BEGINT2:= T_NEW;DRA2:= DRA_NEW END
                 ELSE BEGIN T1:= T_NEW;DRA1:= DRA_NEW END;
   UNTIL (ABS(DRA_NEW)<EPS);
   T_CONJ:= T_NEW;
   (* проверка, падает ли тень Луны на Землю                      *)
   DE_CON:=T_EVAL(DEPOLY,T_CONJ);
```

```
    R_CON:= T_EVAL(RPOLY, T_CONJ);
    CONJ := (ABS(SN(DE_CON-DE_STAR)*R_CON) < 1.5);
  END;
END;
(*--------------------------------------------------------------*)
```

8.3. Фундаментальная плоскость

В параллельных лучах света звезды Луна отбрасывает тень, которая в виде длинного цилиндра простирается от Луны до Земли. На любой плоскости, перпендикулярной оси тени, она рисует круг с диаметром, равным диаметру Луны. Такая плоскость, проходящая через центр Земли, называется *фундаментальной плоскостью*. В проекции на фундаментальную плоскость особенно удобно описывать положение наблюдателя относительно тени Луны.

Если центр прямоугольной системы координат поместить в центр Земли (рис.8.3), а систему расположить так, чтобы ось z указывала на звезду с координатами (α_*, δ_*), а ось x лежала в экваториальной плоскости, то оси x и y определят фундаментальную плоскость. Точка с прямым восхождением α и склонением δ на расстоянии r от Земли в этой системе имеет координаты

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = r \cdot \begin{bmatrix} -\cos\delta \ \sin(\alpha - \alpha_*) \\ \sin\delta\cos\delta_* - \cos\delta \ \sin\delta_* \cos(\alpha - \alpha_*) \\ \sin\delta\sin\delta_* + \cos\delta \ \cos\delta_* \cos(\alpha - \alpha_*) \end{bmatrix}. \tag{8.3}$$

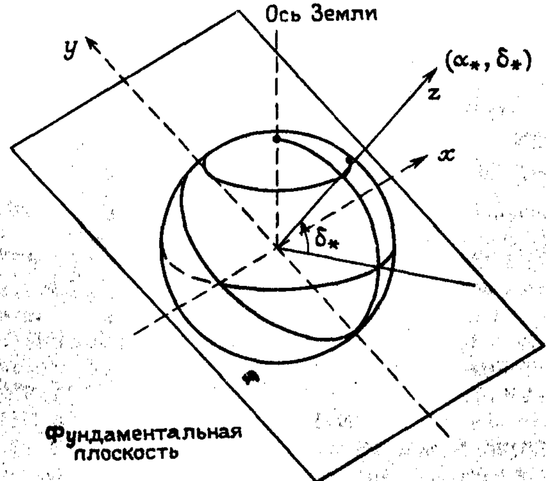


Рис.8.3. Фундаментальная плоскость.

Если вместо (α, δ, r) взять координаты Луны (α_M, δ_M, r_M), x_M и y_M будут координатами середины тени Луны на фундаментальной плоскости.

Таким же образом, из экваториальных координат наблюдателя можно получить точку (x_0, y_0) на фундаментальной плоскости, в которую проецируется луч звезды, проходящий через наблюдателя. Склонение места наблюдения описывает при этом угол между геоцентрическим вектором этого пункта и плоскостью земного и небесного экватора. Ввиду этого его называют также *геоцентрической* широтой φ', которую нельзя путать с *географической* широтой φ. Последняя используется гораздо чаще и представляет собой угол между земной осью и местным горизонтом и, в частности, является мерой высоты Полярной звезды над горизонтом. Причиной различия φ' и φ является сплюснутость Земли (разд.7.5). Из-за суточного вращения Земля имеет форму не шара, а эллипсоида вращения. В то время как радиус Земли на экваторе равен $r_\oplus$ = 6378,14 км, расстояние полюса от центра Земли примерно на 20 км меньше. Отношение этой разности к радиусу Земли называют сплюснутостью Земли: $f \approx 1/298$[1]. Обычно значение геоцентрической широты места неизвестно, и в вычислениях пользуются географической широтой. Ее можно найти по существующим картам и атласам. Подпрограмма SITE использует формулы (7.8) для определения значений $r \cos \varphi'$ и $r \sin \varphi'$, необходимых в (8.3).

```
(*-----------------------------------------------------------------*)
(* SITE:   вычисление геоцентрических координат по географическим   *)
(*         RCPHI: r * cos(phi') (геоцентрич. в радиусах Земли)       *)
(*         RSPHI: r * sin(phi') (геоцентрич. в радиусах Земли)       *)
(*         PHI  : географическая широта (град.)                      *)
(*-----------------------------------------------------------------*)
PROCEDURE SITE ( PHI: REAL; VAR RCPHI,RSPHI: REAL );
  CONST E2 = 0.006694; (* e**2=f(2-f) при сплюснутости Земли       *)
                       (*    f=1/298.257                           *)
  VAR   N,SNPHI: REAL;
  BEGIN
    SNPHI:= SN(PHI);   N    := 1.0/SQRT(1.0-E2*SNPHI*SNPHI);
    RCPHI:= N*CS(PHI); RSPHI:= (1.0-E2)*N*SNPHI;
  END;
(*-----------------------------------------------------------------*)
```

Прямое восхождение места наблюдения идентично прямому восхождению звезды, которую наблюдатель видит точно в меридиане, и равно местному звездному времени. Его можно вычислить с помощью выражений (3.4) и (3.5). Теперь нам необходимо еще раз обратиться к различным системам измерения времени в астрономии. При вычислении положения Луны до сих пор молчаливо подразумевалось, что все моменты заданы в единицах *эфемеридного времени* ET (= динамическое время TDB/TDT). Эта физически равномерная мера времени применяется во всех случаях

[1] Традиционно f обозначает как сплюснутость, так и одну из координат на фундаментальной плоскости. Путаница, однако, маловероятна.

вычисления орбит небесных тел, о чем свидетельствует и название «эфемеридное время». Вычисление звездного времени (при помощи подпрограммы LMST из разд.3.3), напротив, требует знания *всемирного времени* UT, которое в общем соответствует нашему обычному времени (с точностью до часовых поясов). Разность ET и UT составляет около минуты и может быть взята из табл.3.1. В разд.7.3 была приведена программа ETMINUT, которая посредством аппроксимации полиномом позволяет вычислить искомую величину в интервале между 1900 и 1985 гг. Данные для более позднего интервала можно найти в различных астрономических ежегодниках Так как разность между всемирным и эфемеридным временем меняется медленно, вводить ее в машину нужно один раз в начале работы.

8.4. Начало и конец покрытия

Ход покрытия зависит от разностей

$$f = x_0 - x_M ,$$

$$g = y_0 - y_M$$

координат Луны и наблюдателя на фундаментальной плоскости. Если расстояние от центра тени Луны до наблюдателя $\sqrt{f^2 + g^2}$ меньше радиуса Луны R_M, то в данном месте покрытие состоится. Момент начала и конца покрытия получают из условия

$$f(t)^2 + g(t)^2 = R_M^2 = k^2 R_\oplus^2 , \tag{8.4}$$

где

$$k = R_M / R_\oplus = 0,2725$$

—отношение радиуса Луны к радиусу Земли.

В своем движении вокруг Земли Луна проходит расстояние, равное диаметру Земли, примерно за 4 ч. За это же время проходит по поверхности Земли и тень Луны Чтобы установить моменты контактов достаточно начать расчет с момента за два с половиной часа до соединения. Чтобы определить оба момента, удовлетворяющих условию (8.4), нужно на каждые 15 мин вычислять значение

$$s(t) = f^2(t) + g^2(t) - k^2 R_\oplus^2 .$$

В начале наблюдений s положительно, так как место наблюдения находится еще вне тени Луны Когда s становится отрицательным, звезда скрывается за краем Луны По трем последовательным значениям s_0, s_+ можно определить параболу, которая в хорошем приближении описывает ход $s(t)$ в течение получаса Если за это время s один или два раза обращается в нуль, то корни могут быть определены решением квадратного уравнения. В противном случае выбирают три следующих значения s и поступают с ними таким же образом. Этот метод полностью аналогичен определению времени восхода и захода светил в гл.3. Там уже была представлена процедура QUAD для квадратичной интерполяции и определения нулей функции, заданной в трех точках. Она используется в подпрограмме SHADOW для определения начала и конца покрытия. Полный текст этой подпрограммы дается в конце главы.

Предвычисление покрытий помимо моментов контакта включает и другие величины, представляющие интерес для наблюдателей: позиционный угол начала и конца покрытия и коэффициенты места наблюдения. Определение позиционного угла представлено на рис.8.4. Это угол между направлением на север и линией, соединяющей центр Луны и точку покрытия или открытия на лимбе Луны Позиционный угол измеряется против часовой стрелки от 0 до 360°. Значение позиционного угла ϑ зависит только от координат на фундаментальной плоскости f и g в моменты начала или конца покрытия и может быть вычислено при помощи зависимостей

$$\cos\vartheta = -g/\sqrt{f^2+g^2}\,,$$
$$\sin\vartheta = -f/\sqrt{f^2+g^2}\,. \tag{8.5}$$

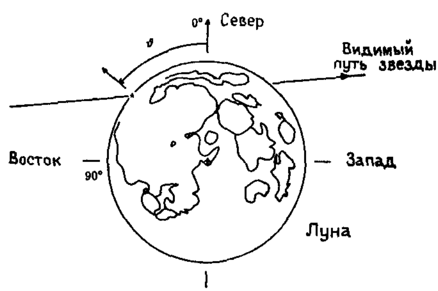


Рис.8.4. Позиционный угол звезды в момент контакта с краем Луны.

Коэффициенты места наблюдения — это величины, приближенно описывающие зависимость времени контакта от места наблюдения. Их используют прежде всего для того, чтобы определить моменты контакта в данном месте по моментам контакта, предсказанным для соседнего места Коэффициент a описывает, как меняется момент контакта t в зависимости от географической долготы λ, а коэффициент b — отражает такую же зависимость для географической широты φ:

$$a = \frac{dt}{d\lambda}\,, \qquad b = \frac{dt}{d\varphi}\,.$$

Типичные значения обоих коэффициентов равны примерно $1^m/°$. Однако при касательных покрытиях, когда начало и конец очень близки друг другу, коэффициенты места наблюдения заметно увеличиваются. Уравнения для вычисления коэффициентов места наблюдения содержат в основном уже известные величины:

$$a = K\,[r \cos \varphi' \,(f \cos (\alpha_B - \alpha_*) + g \sin \delta_* \sin (\alpha_0 - \alpha_*))],$$

$$b = K\,[r \sin \varphi' \,(f \sin (\alpha_B - \alpha_*) + g \cos \delta_* \sin (\alpha_0 - \alpha_*)) - \\ - r \cos \varphi' \, g \cos \delta_*], \qquad (8.6)$$

где

$$K = -\frac{1{,}047\ ^m/°}{(f\dot{f} + g\dot{g}) \cdot 1^h}.$$

Здесь $\dot{f}$ и $\dot{g}$—производные от f и g по времени, которые для простоты можно представить в виде следующих выражений:

$$\dot{f}(t) \approx \frac{f(t + 0{,}25^h) - f(t)}{0{,}25^h}, \qquad \dot{g}(t) \approx \frac{g(t + 0{,}25^h) - g(t)}{0{,}25^h},$$

где t—момент покрытия или открытия. При вычислении коэффициентов места наблюдения разницей между географической (φ) и геоцентрической (φ') широтой можно пренебречь.

8.5. Программа OCCULT

Программа OCCULT позволяет вычислить покрытия звезд Луной для определенного промежутка времени в определенном месте наблюдения Вычисляются начало и конец покрытия краем Луны, соответствующий позиционный угол, высота Луны над горизонтом в момент контакта и коэффициенты места наблюдения, с помощью которых можно найти моменты контактов в соседних местах.

Координаты звезд, которые исследуются на предмет покрытия, представляются в виде файла данных с именем OCCINP(.DAT). Первая строка этого файла содержит эпоху и равноденствие использованного каталога (как правило, 1950.0 или 2000 0). Эти данные служат для учета собственных движений и пересчета положений звезд на текущее равноденствие. Наконец, для каждой звезды в отдельной строке задаются

* прямое восхождение ($^{h\ m\ s}$),
* собственное движение по прямому восхождению за сто лет (^{s}),
* склонение (° ′ ″),
* собственное движение по склонению (″) и
* имя звезды.

Длина имени ограничивается максимум 17 буквами, но при необходимости ее можно увеличить изменением константы NAME_LENGTH. Следующий пример построения файла данных для Плеяд взят из Зодиакального каталога (см табл.8.1)

1950.0				1950.0				
3	41	49.540	+0 110	+24	8	1 58	-4.26	Целено
3	41	54.063	+0 144	+23	57	27.90	-4.16	Электра
3	42	13 603	+0 171	+24	18	43 07	-4.20	Тайгета
3	42	50.749	+0.127	+24	12	46 74	-4.78	Майя
3	42	55.387	+0.100	+24	23	59.69	-4.91	Астеропа
3	43	21.195	+0.145	+23	47	38 98	-4.34	Меропа
3	44	30 427	+0.144	+23	57	7.52	-4.47	Альциона
3	46	11.022	+0 111	+23	54	7.41	-4.71	Атлас
3	46	12.393	+0.114	+23	59	7.18	-5.53	Плейона

В начале работы программой запрашиваются географические координаты места наблюдения и временной интервал, в котором ищут покрытия. Весь интервал времени обрабатывается шагами по 10 сут, в которых координаты Луны вначале представляются полиномами Чебышева. Для каждой звезды из массива данных после вычисления видимых координат (подпрограмма APPARENT) исследуется вероятность покрытия звезды Луной (подпрограмма EXAMINE). Уже описанная подпрограмма CONJUNCT определяет момент соединения Луны и звезды и проверяет расстояние оси тени от центра Земли Подпрограмма SHADOW для данного места наблюдения определяет моменты начала и конца покрытия, а также соответствующие им позиционные углы и коэффициенты места наблюдения. Из предвычисленных таким образом покрытий выделяются те, которые происходят по меньшей мере на 5° выше горизонта в достаточной темноте. Для этого подпрограмма DARKNESS проверяет, находится ли Солнце к середине покрытия по меньшей мере на 6° под горизонтом (гражданские сумерки). В отличие от распространенных методов предвычисления покрытий, не выясняется, на каком крае Луны, освещенном или не освещенном, происходит контакт Нужно упомянуть, что хронологическая последовательность выдачи покрытий необязательна, она зависит только от последовательности ввода звездных координат.

```
(*-----------------------------------------------------------*)
(*                          OCCULT                            *)
(*          Предвычисление покрытий звезд Луной               *)
(*                    Версия 21.12.1988                       *)
(*-----------------------------------------------------------*)
PROGRAM OCCULT(INPUT,OUTPUT,OCCINP);
  CONST MAX_TP_DEG=13;             (* порядок полинома Чебышева *)
        TOVLAP   =  3.42E-6;      (*  3h в юлианских столетиях  *)
        T_SEARCH = 2.737850787E-4;(* 10d в юл. столетиях        *)
        NAME_LENGTH= 17;       (* максимальная длина имени звезды *)
  TYPE  REAL33      = ARRAY[1..3,1..3] OF REAL;                *)
        NAME_STRING= ARRAY[1..NAME_LENGTH] OF CHAR;
```

```
        TPOLYNOM  = RECORD   (* полином Чебышева            *)
                  M : INTEGER; (*      степень                *)
                  A,B: REAL;   (*      интервал               *)
                  C  : ARRAY [0..MAX_TP_DEG] OF REAL (* коэфф. *)
                END;
   VAR  T_BEGIN,T_END,T1,T2,TM,T_EQX,T_EPOCH: REAL;
        ETDIFUT                             : REAL;
        RA_STAR,DE_STAR                     : REAL;
        VX,VY,VZ                            : REAL;
        LAMBDA,PHI,RCPHI,RSPHI              : REAL;
        RAPOLY,DEPOLY,RPOLY                 : TPOLYNOM;
        PNMAT                               : REAL33;
        OCCINP                              : TEXT;
        NAME                                : NAME_STRING;
(* Сюда вводятся следующие подпрограммы в указанном порядке:     *)
(* SN, CS, ASN, ATN, ATN2, CART, POLAR, GGG, GMS, T_EVAL         *)
(* T_FIT_LBR,QAD,ECLEQU,ABERRAT,SITE,PMATEQU,PRECART,NUTEQU      *)
(* PNMATRIX,MJD,CALDAT,LMST,ETMINUT,MOON,MOONEQU,T_FIT_MOON*)
(* MINISUN, CONJUNCT                                             *)
(*-------------------------------------------------------------  *)

(*-------------------------------------------------------------  *)
(* GET_INPUT: Ввод временного интервала поиска и координат       *)
(*            наблюдателя                                        *)
(*---------------------------------------------------------------*)
PROCEDURE GET_INPUT(VAR T_BEGIN,T_END,ETDIFUT,LAMBDA,PHI:REAL);
  VAR D,M,Y: INTEGER;
      T    : REAL;
      VALID: BOOLEAN;
  BEGIN
    WRITELN;
    WRITELN('      OCCULT: Покрытия звезд Луной
    WRITELN(             Версия 21.12.88
    WRITELN('      (c) 1988 Thomas Pfleger, Oliver Montenbruck
    WRITELN;
    WRITE  (' начальная дата поиска (DD MM YYYY )    ...
         READLN(D,M,Y);
    T_BEGIN:=   (MJD(D,M,Y,0)-51544.5)/36525.0;
    WRITE  (' конечная дата поиска (DD MM YYYY )     ...
         READLN(D,M,Y);
    T_END  := (MJD(D,M,Y,0)-51544.5)/36525.0;
    T      := ( T_BEGIN + T_END ) / 2.0;
    ETMINUT  ( T, ETDIFUT, VALID);
    IF  (VALID)
      THEN WRITE(' разность ET-UT (предложение:',
                  TRUNC(ETDIFUT+0.5):3,' c)          ...            ');
      ELSE WRITE(' разность ET-UT (c)                ...            ');
```

```
    READLN(ETDIFUT);
WRITE ('коорд. наблюдателя  долгота (полож.к зап.)       ...         ');
      READLN(LAMBDA);
WRITE ('                      широта                     ...         ');
      READLN(PHI);
  END,

(*---------------------------------------------------------------   *)
(*     HEADER: заголовки при печати                                 *)
(*---------------------------------------------------------------   *)
PROCEDURE HEADER;
  BEGIN
    WRITELN;
WRITELN (' Дата время(UT) H/K     Поз.уг.     h    a    b     звезда');
WRITELN ('                h  m  s            o      o    m    m        ');
  END;

(*---------------------------------------------------------------   *)
(* GETSTAR  чтение координат звезд из файла данных OCCINP и         *)
(*          учет собственных движений                               *)
(*---------------------------------------------------------------   *)
PROCEDURE GETSTAR ( T_EPOCH,T. REAL;
                      VAR RA,DEC: REAL; VAR NAME: NAME_STRING);
  VAR G,M,I               · INTEGER;
      S, PM_RA, PM_DEC:  REAL;
  BEGIN
 READ(OCCINP,G,M,S); GGG(G,M,S,RA); (* прямое восхожд. в эпоху      *)
    READ(OCCINP,PM_RA );
 READ(OCCINP,G,M,S); GGG(G,M,S,DEC);(* склонение в эпоху            *)
   READ(OCCINP,PM_DEC);
 RA  :=RA+(T-T_EPOCH)*PM_RA/3600.0;  (* собственное движение         *)
                                     (* по прямому восхождению       *)
 DEC:=DEC+(T-T_EPOCH)*PM_DEC/3600.0;(* собственное движение          *)
                                     (* по склонению                 *)
 FOR I:=1 TO NAME_LENGTH DO          (* название звезды              *)
 IF (NOT EOLN(OCCINP) ) THEN READ(OCCINP,NAME[I])
    ELSE  NAME[I]·=' ';
    READLN(OCCINP);
  RA:= 15.0 * RA;                    (* RA (град.)                   *)
  END;

(*---------------------------------------------------------------   *)
(* APPARENT  : видимые координаты звезды                            *)
(*    PNMAT  : матрица прецессии и нутации                          *)
(*    VX,VY,VZ: скорость Земли (экваториальная в единицах с)        *)
(*    RA,DEC : прямое восхождение и склонение                       *)
(*---------------------------------------------------------------   *)
```

```
PROCEDURE APPARENT ( PNMAT:REAL33; VX,VY,VZ: REAL;
                    VAR RA,DEC: REAL );
  VAR X,Y,Z,R: REAL;
  BEGIN
 CART ( 1.0, DEC,RA, X,Y,Z );      (* прямоугольные координаты звезды  *)
 PRECART ( PNMAT , X,Y,Z );        (* учет прецессии и нутации         *)
 X:=X+VX; Y:=Y+VY; Z:= Z+VZ;       (* аберрация                        *)
 POLAR( X,Y,Z, R,DEC,RA );         (* новые поляр. коор. RA и DEC      *)
  END;
(*-------------------------------------------------------------------*)
(*   SHADOW:                                                          *)
(*                                                                    *)
(* определяет (исходя из момента соединения для конкретного           *)
(* места наблюдений) моменты покрытия и открытия, позиционные         *)
(* углы и коэффициенты места наблюдения                               *)
(*                                                                    *)
(* RAPOLY,DEPOLY,RPOLY: полиномы Чебышева для координат Луны          *)
(* T_CONJ_ET          : момент соединения по RA;                      *)
(*                      юлианские столетия ET с J2000                 *)
(* ETDIFUT            : ET-UT (c)                                     *)
(* LAMBDA,RCPHI,RSPHI : географ координаты наблюдателя                *)
(* RA_STAR,DE_STAR    : координаты звезды                             *)
(* EVENT              : покрытие есть/нет                             *)
(*                      (TRUE/FALSE)                                  *)
(* MJD_UT_IN, MJD_UT_OUT.мом. контакта (мод. юлиан. дата UT)          *)
(* POS_IN, POS_OUT     : позиционный угол                             *)
(* H_IN, H_OUT         : высота звезды над горизонтом                 *)
(* A_IN, A_OUT         : коэффициент места для геогр. долготы         *)
(* B_IN, B_OUT         : коэффициент места для геогр. широты          *)
(*-------------------------------------------------------------------*)

PROCEDURE SHADOW

 ( RAPOLY, DEPOLY, RPOLY                                : TPOLYNOM;
 T_CONJ_ET,ETDIFUT, LAMBDA,RCPHI,RSPHI, RA_STAR,DE_STAR: REAL;
 VAR EVENT                                              : BOOLEAN;
     VAR MJD_UT_IN, MJD_UT_OUT                          : REAL;
 VAR POS_IN,POS_OUT, H_IN,H_OUT, A_IN,A_OUT, B_IN,B_OUT: REAL    );


 CONST DTAB = 0.25,          (* шаг поиска в часах                  *)
       RANGE= 2.25,          (* интервал поиска = +/- (RANGE+DTAB)
                             (* в часах                             *)
       K     =     0.2725;   (* отношение радиус Луны /радиус Земли *)
       CENT  =  876600.0,    (* часов в юлианском столетии          *)
       SID   = 1 0027379,    (* отношение солнечного времени к      *)
                             (* звездному времени                   *)
```

```
VAR I, NZ, NFOUND                     : INTEGER;
    K_SQR, MJD_CONJ_UT, HOUR, F, G: REAL;
    THETA_CONJ, CSDEST, SNDEST  : REAL;
    S_MINUS, S_0, S_PLUS, XE, YE : REAL;
    Z, TIME                     : ARRAY[1..2] OF REAL;

(* FG: координаты f,g на фундаментальной плоскости                *)
PROCEDURE FG ( HOUR: REAL; VAR F,G: REAL);
  VAR T,DEM,RM,RCDEM,RSDEM,DRAM,DRA: REAL;
  BEGIN
   T    := T_CONJ_ET + HOUR/CENT;
   DEM := T_EVAL(DEPOLY,T);    RM   := T_EVAL(RPOLY,T);
   RCDEM:=RM * CS(DEM);        RSDEM:=RM * SN(DEM);
   DRAM:= T_EVAL(RAPOLY,T)     - RA_STAR;
   DRA  := 15.0 * ( THETA_CONJ + HOUR*SID ) - RA_STAR;
   F    := +RCDEM*SN(DRAM)     - RCPHI*SN(DRA) ;
   G    := + RSDEM*CSDEST      - RCDEM*SNDEST*CS(DRAM)
           - RSPHI*CSDEST      + RCPHI*SNDEST*CS(DRA);
  END;

(* CONTACT: позиционный угол, высота над горизонтом,               *)
           (* коэффициенты места наблюдения                         *)
PROCEDURE CONTACT ( HOUR: REAL; VAR POS,H,A,B: REAL );
  VAR F,G,FF,GG,DF,DG,FAC,DRA,CDRA,SDRA: REAL;
  BEGIN
    FG ( HOUR, F, G );  FG ( HOUR+DTAB , FF, GG );
    DF  := (FF-F)/DTAB;   DG:= (GG-G)/DTAB;
    POS := ATN2(-F,-G);   IF POS<0.0 THEN POS:=POS+360.0;
    FAC := 1.047 / (F*DF+G*DG);
    DRA := 15.0 * ( THETA_CONJ + HOUR*SID ) - RA_STAR;
    CDRA:= CS(DRA);  SDRA := SN(DRA);
    A   := -FAC * RCPHI * ( F*CDRA + G*SDRA*SNDEST );
    B   := -FAC*(RSPHI * (F*SDRA-G*SNDEST*CDRA)-RCPHI*G*CSDEST);
    H   := ASN ( RSPHI*SNDEST + RCPHI*CSDEST*CDRA );
 END;

BEGIN

(* модифицированные юлианские даты и звездное время соединения  *)
 MJD_CONJ_UT:=T_CONJ_ET*36525.0 + 51544.5 - ETDIFUT/86400.0;
 THETA_CONJ:= LMST ( MJD_CONJ_UT, LAMBDA );

 (* вспомогательные величины                                      *)
 K_SQR:= K*K;    CSDEST:= CS(DE_STAR);  SNDEST:= SN(DE_STAR);

 (* поиск момента контакта                                        *)
```

```
    NFOUND:= 0;  TIME[1]:= 0.0;  TIME[2]:= 0.0;
    HOUR  := -RANGE-2.0*DTAB;
    FG (-RANGE-DTAB,F,G); S_PLUS := F*F+G*G-K_SQR;
    REPEAT
      HOUR   := HOUR + 2.0*DTAB;
      S_MINUS:= S_PLUS;
      FG ( HOUR   ,F,G );      S_0     := F*F+G*G-K_SQR;
      FG ( HOUR+DTAB,F,G );S_PLUS := F*F+G*G-K_SQR;
      QUAD ( S_MINUS,S_0,S_PLUS, XE,YE, Z[1],Z[2],NZ );
      FOR  I:= 1 TO NZ DO TIME[NFOUND+I]:= HOUR+DTAB*Z[I];
      NFOUND:= NFOUND + NZ; EVENT      := (NFOUND=2);
    UNTIL ( (EVENT) OR (HOUR>=RANGE) ) ;

    (* вычисление условий покрытия, если оно происходит            *)
    IF EVENT THEN
      BEGIN
        MJD_UT_IN := MJD_CONJ_UT+ TIME[1] / 24.0;
        MJD_UT_OUT:= MJD_CONJ_UT+ TIME[2] / 24.0;
        CONTACT ( TIME[1], POS_IN, H_IN, A_IN, B_IN );
        CONTACT ( TIME[2], POS_OUT,H_OUT,A_OUT,B_OUT );
      END,

  END;

(*-----------------------------------------------------------*)
(* DARKNESS       : тест на гражданские сумерки                *)
(*      MODJD     : модифицированная юлианская дата            *)
(*      LAMBDA    : геогр. долгота (>0 к западу от Гринвича)   *)
(*      CPHI,SPHI: sin и cos географической широты            *)
(*-----------------------------------------------------------*)
FUNCTION DARKNESS ( MODJD, LAMBDA,CPHI,SPHI: REAL ): BOOLEAN;
  VAR T,RA,DEC,TAU,SIN_HSUN: REAL;
  BEGIN
    T         := (MODJD-51544 5)/36525.0;
    MINI_SUN  (T,RA,DEC);
    TAU       := 15.0 * (LMST(MODJD,LAMBDA) - RA);
    SIN_HSUN:= SPHI*SN(DEC) + CPHI*CS(DEC)*CS(TAU);
    DARKNESS:= ( SIN_HSUN < -0.10 );
  END;

(*-----------------------------------------------------------*)
(*  EXAMINE:                                                    *)
(*  исследует, состоится ли покрытие, вычисляет даты и выдает их  *)
(*                                                              *)
(*  T1,T2              : интервал поиска в юл. столет. с J2000   *)
(*  RAPOLY,DEPOLY,RPOLY:полиномы Чебышева для координат Луны     *)
(*  ETDIFUT            : ET-UT (c)                              *)
(*  LAMBDA,RCPHI,RSPHI: географические координаты наблюдателя    *)
```

```
(*   RA_STAR,DE_STAR. координаты звезды                              *)
(*   NAME           : название звезды                                *)
(*-------------------------------------------------------------------*)

PROCEDURE EXAMINE ( T1,T2: REAL; RAPOLY,DEPOLY,RPOLY: TPOLYNOM;
                    ETDIFUT,LAMBDA,RCPHI,RSPHI,RA_STAR,DE_STAR: REAL;
                        NAME: NAME_STRING);

 CONST H_MIN=5.0; (* минимальная высота над горизонтом (град)      *)

  VAR DAY,MONTH,YEAR,H,M,I                               : INTEGER;
      S,HOUR, T_CONJ_ET, MJD_UT_IN, MJD_UT_OUT           : REAL;
      POS_IN,POS_OUT, H_IN,H_OUT, A_IN,A_OUT, B_IN,B_OUT: REAL;
      CONJ, TAKES_PLACE                                  : BOOLEAN;

  BEGIN

   (* тест на соединение по RA и определение момента соединения    *)

   CONJUNCT ( T1,T2, RAPOLY,DEPOLY,RPOLY,
              RA_STAR,DE_STAR, CONJ, T_CONJ_ET );

   IF CONJ THEN

     BEGIN

   (* поиск возможных покрытий в заданном месте и вычисление       *)
   (* моментов контактов, высоты над горизонтом                    *)
   (* и коэффициентов места наблюдения                             *)

       SHADOW ( RAPOLY,DEPOLY,RPOLY, T_CONJ_ET, ETDIFUT,
                LAMBDA,RCPHI,RSPHI, RA_STAR, DE_STAR,
                TAKES_PLACE, MJD_UT_IN, MJD_UT_OUT, POS_IN,
                POS_OUT, H_IN,H_OUT, A_IN,A_OUT, B_IN,B_OUT );

    (* вывод результатов, если покрытие происходит                  *)
    (* в темное время и достаточно высоко над горизонтом            *)

       IF TAKES_PLACE THEN
       IF ( (H_IN>H_MIN) OR (H_OUT>H_MIN) ) THEN
       IF DARKNESS ( (MJD_UT_IN+MJD_UT_OUT)/2.0,
                     LAMBDA,RCPHI,RSPHI)  THEN

         BEGIN

         (* начало покрытия                                         *)
         CALDAT ( MJD_UT_IN, DAY,MONTH,YEAR,HOUR );
```

```
                GMS (HOUR,H,M,S);
          WRITE     ( DAY:4,'.', MONTH:2, '. ', H.3, M:3,
                    TRUNC(S+0.5):3, ' H ', TRUNC(POS_IN+0.5):5,
                    TRUNC(H_IN+0.5):6, A_IN:8:1,
                          B_IN:6:1,' ':3          );
          FOR I:=1 TO NAME_LENGTH DO WRITE(NAME[I]); WRITELN;

         (* конец покрытия                                    *)
  CALDAT ( MJD_UT_OUT, DAY,MONTH,YEAR,HOUR);   GMS (HOUR,H,M,S);
         WRITELN ( DAY:4,'.', MONTH:2, '.  ', H:3, M:3,
                    TRUNC(S+0.5):3, '   K  ', TRUNC(POS_OUT+0.5):5,
                    TRUNC(H_OUT+0.5):6, A_OUT:8:1, B_OUT:6:1      );

        END;

     END;

  END;
(*------------------------------------------------------------*)

BEGIN (* основная программа                                   *)

 (* считывание интервала поиска и координат наблюдателя      *)

  GET_INPUT ( T_BEGIN,T_END, ETDIFUT, LAMBDA,PHI );

  (* вычисление геоцентрических координат наблюдателя         *)

  SITE ( PHI, RCPHI,RSPHI );

  (* поиск покрытий в последовательных интервалах             *)
  T2 .= T_BEGIN;

  REPEAT
  T1 :=T2;    T2:=T1+T_SEARCH;

  (* аппроксимация координат Луны полиномами Чебышева         *)

T_FIT_MOON(T1-TOVLAP,T2+TOVLAP,MAX_TP_DEG,RAPOLY,DEPOLY,RPOLY);

    (* вывод заголовков

    HEADER,

    (* открытие файла звездных данных, считывание эпохи и      *)
    (* равноденствия                                           *)
 RESET ( OCCINP );      (* Standard Pascal                     *)
```

```
(* ASSIGN (OCCINP,'OCCINP.DAT'); RESET(OCCINP);                   *)
                                      (* TURBO Pascal             *)
(* RESET ( OCCINP,'OCCINP.DAT'); *)   (* ST Pascal plus           *)

   READLN   ( OCCINP, T_EPOCH, T_EQX );
   T_EQX := ( T_EQX  -2000.0 ) / 100.0;
   T_EPOCH:=( T_EPOCH-2000.0 ) / 100.0;

   (* матрица преобразования для перехода от среднего            *)
   (* равноденствия каталога к истинному равноденствию           *)
   (* даты, соответствующей середине интервала поиска            *)
   TM:=  (T1+T2)/2 0;
   PN_MATRIX ( T_EQX, TM, PNMAT );

   (* гелиоцентрическая скорость Земли для учета аберрации       *)
   ABERRAT ( TM, VX,VY,VZ );

   (* обработка списка звезд и поиск возможных покрытий          *)
   WHILE NOT EOF(OCCINP) DO
    BEGIN
      (* считывание новых координат звезд                        *)
      GETSTAR ( T_EPOCH, TM, RA_STAR,DE_STAR,NAME );
      (* вычисление видимых координат                            *)
      APPARENT( PNMAT, VX,VY,VZ, RA_STAR,DE_STAR );
      (* поиск возможных покрытий                                *)
      EXAMINE ( T1,T2, RAPOLY,DEPOLY,RPOLY,
              ETDIFUT, LAMBDA,RCPHI,RSPHI, RA_STAR,DE_STAR,NAME );
    END;
  WRITELN;
 UNTIL (T2 >= T_END);
END.
(*-----------------------------------------------------------    *)
```

В качестве примера мы с помощью программы OCCULT вычислим две серии покрытий Плеяд, случившихся в 1989 г. В качестве интервала поиска выберем промежуток между 15 сентября и 15 ноября 1989 г. Вычисления проведены для Мюнхена, который расположен на 11,6° в. д. и 48,1° с. ш. Мы предполагаем, что разность эфемеридного и всемирного времен неизвестна (как это делают в предсказаниях покрытий) и принимаем в качестве оценки значение 56 с. Свое значение величины ΔT = ET−UT программа OCCULT может дать только в интервале 1900−1985 гг

Заданный промежуток времени исследуется шагами по 10 дней, в пределах которых и выдаются все найденные покрытия. В последующем списке выдача запрещена, если покрытие не происходит. Курсив применен для обозначения вводимых данных.

OCCULT: Покрытия звезд Луной
Версия 21.12.88
(с) 1988 Thomas Pfleger, Oliver Montenbruck

начальная дата поиска (DD MM YYYY) ... *15 09 1989*
конечная дата поиска (DD MM YYYY) ... *15 11 1989*
разность ET—UT (с) ... *56.00*
коорд. наблюдателя: долгота (полож.к зап.) ... *-11.60*
широта ... *48.10*

Дата	Время (UT) h m s	Н/К	Поз. уг. °	h °	a m	b m	Звезда
19. 9.	21 49 45	Н	96	25	−0.3	1.3	Целено
19. 9	22 41 3	К	222	34	−0.0	2.1	
19 9.	22 1 60	Н	67	27	−0.1	1.7	Тайгета
19. 9.	23 0 40	К	251	37	−0.5	1.7	
19. 9.	22 15 35	Н	98	30	−0.5	1.3	Майя
19. 9.	23 7 34	К	220	38	−0 1	2.3	
19. 9.	22 21 1	Н	62	31	−0.1	1.8	Астеропа
19. 9.	23 20 38	К	256	41	−0.6	1.6	

... . .

Дата	Время (UT) h m s	Н/К	Поз. уг. °	h °	a m	b m	Звезда
13.11.	18 3 47	Н	36	24	0.3	2.0	Целено
13.11.	18 48 58	К	285	31	−0.7	1.1	
13.11.	17 54 51	Н	73	22	0.0	1.5	Электра
13.11.	18 49 35	К	248	31	−0.2	1.7	
13.11.	18 27 22	Н	37	28	0 2	2.1	Майя
13 11.	19 14 37	К	284	35	−0.8	1.1	
13 11.	18 43 45	Н	153	30	−5 0	−4 8	Меропа
13.11.	18 50 46	К	168	31	4.2	7.7	
13.11.	19 0 9	Н	122	33	−1 1	0 6	Альциона
13.11.	19 36 7	К	198	39	0.4	2.9	

9. ОПРЕДЕЛЕНИЕ ОРБИТЫ

Классическая задача определения орбиты состоит в получении орбитальных элементов планет, комет или астероидов из возможно меньшего числа наблюдений. Определение орбиты является обратной задачей по отношению к вычислению эфемерид — положений по известным элементам орбиты Каждое наблюдение, выполненное с Земли в определенный момент, дает две сферические координаты. Они могут быть измерены относительно небесного экватора (прямое восхождение и склонение) или эклиптики (эклиптическая долгота и широта). Расстояние при этом измерить невозможно, поэтому для определения орбиты эта величина не может быть использована. Для определения шести элементов орбиты необходимо получить столько же независимых значений, т. е. требуется три наблюдения.

Представленный здесь метод Буцериуса определения орбит, в сущности, восходит к Гауссу, но в некоторых отношениях заметно проще. Эти упрощения накладывают определенные ограничения на его использование, но, с другой стороны, делают его более понятным и легко применимым.

9.1. Определение орбиты по двум векторам положения

Орбитальные элементы используются для описания орбит планет и комет прежде всего потому, что с их помощью можно получить четкую интерпретацию индивидуальных величин. Однако для определения орбиты удобнее использовать другой метод описания. Орбита задается в одинаковой степени однозначно, если известно положение и скорость небесного тела в какой-то момент времени или же две точки на орбите и моменты, к которым они относятся. В то время как в первом случае для определения орбиты используют метод Лапласа, описание орбиты двумя векторами положения используется в методе Гаусса. Задача этого раздела состоит в изложении метода вычисления элементов орбиты, заданной двумя положениями $\mathbf{r}_a$ и $\mathbf{r}_b$ и соответствующими им моментами t_a и t_b. Задача определения орбиты сводится при этом к задаче определения двух гелиоцентрических положений по трем наблюдаемым геоцентрическим направлениям

Первый промежуточный шаг состоит в вычислении *отношения площади сектора к площади треугольника,* что является самым сложным этапом вычисления элементов орбиты Как раз эта величина играет главную роль для дальнейшего хода определения орбиты, но это станет ясно лишь в следующем разделе.

9.1.1. Отношение площадей сектора и треугольника

Площадь треугольника Δ, образованного векторами $\mathbf{r}_a$ и $\mathbf{r}_b$ (рис.9.1), зависит от длины сторон r_a и r_b и угла между ними $\nu_b - \nu_a$, который всегда должен быть меньше 180°:

$$\Delta = 1/2\, r_a\, r_b \sin(\nu_b - \nu_a) . \tag{9.1}$$

Здесь ν_a и ν_b—значения истинной аномалии в крайних точках орбиты.

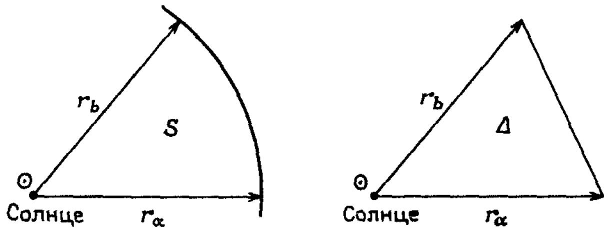


Рис.9.1. Площадь сектора и треугольника.

Площадь сектора S, ограниченного $\mathbf{r}_a$, $\mathbf{r}_b$ и лежащей между ними дугой орбиты, на основании второго закона Кеплера (закон площадей) пропорциональна разности времен t_a и t_b:

$$S = 1/2 \sqrt{GM_\odot} \sqrt{a(1 - e^2)}\, (t_b - t_a) . \tag{9.2}$$

Здесь a и e—соответственно полуось и эксцентриситет орбиты, соединяющей заданные точки (гл.4). Введя еще параметр орбиты $p = a \cdot (1 - e^2)$, для отношения двух площадей получим выражение:

$$\eta = \frac{S}{\Delta} = \frac{\sqrt{p} \cdot \tau}{r_a\, r_b \sin(\nu_b - \nu_a)} , \tag{9.3}$$

где с целью упрощения интервал времени определен выражением

$$\tau = \sqrt{GM_\odot}\,(t_b - t_a) . \tag{9.4}$$

В формуле (9.3) содержится параметр p, который не был предварительно выражен через векторы $\mathbf{r}_a$ и $\mathbf{r}_b$. Пытаясь исключить этот параметр с помощью известных формул из задачи двух тел, мы обнаружим, что в этом случае не удается представить η в явном виде. Вместо этого получим трансцендентное уравнение[1]

[1] Мы не будем приводить вывод этого уравнения, занимающий много страниц. Интересующийся читатель может обратиться к литературе, указанной в приложении.

$$\eta = 1 + \frac{m}{\eta^2} \cdot W\left[\frac{m}{\eta^2} - l\right] \tag{9.5}$$

со вспомогательными (положительными) переменными:

$$m = \frac{\tau^2}{\sqrt{2(r_a r_b + \mathbf{r}_a \cdot \mathbf{r}_b)}^3},$$

$$l = \frac{r_a + r_b}{2\sqrt{2(r_a r_b + \mathbf{r}_a \cdot \mathbf{r}_b)}} - \frac{1}{2}, \tag{9.6}$$

из которого η может быть получено посредством итераций. Функция W при этом задается по частям:

$$W(w) = \begin{cases} \dfrac{2g - \sin(2g)}{\sin^3(g)}, & g = 2\arcsin\sqrt{w}, \quad 0 < w < 1, \\ \dfrac{4}{3} + \dfrac{4\cdot 6}{3\cdot 5}w + \dfrac{4\cdot 6\cdot 8}{3\cdot 5\cdot 7}w^2 + \ldots, & w \approx 0, \\ \dfrac{\operatorname{sh}(2g) - 2g}{\operatorname{sh}^3(g)}, & g = 2\operatorname{arcsh}\sqrt{-w}, \quad w < 0. \end{cases} \tag{9.7}$$

Для вычисления η способом итераций нужно воспользоваться методом секущих. Если мы напишем

$$f(x) = 1 - x + \frac{m}{x^2} W\left[\frac{m}{x^2} - l\right],$$

то искомое значение η будет корнем функции f. Исходя из двух приближенных значений η_{i-1} и η_i, находим уточненное приближение η_{i+1}:

$$\eta_{i+1} = \eta_i - f(\eta_i)\frac{f(\eta_i) - f(\eta_{i-1})}{\eta_i - \eta_{i-1}}.$$

Говоря наглядно, при этом мы ищем нуль секущей, проходящей через точки $(\eta_{i-1}, f(\eta_{i-1}))$ и $(\eta_i, f(\eta_i))$ графика f. При многократном повторении этого шага итерации быстро приведут к искомому значению отношения площадей сектора и треугольника. Хорошие начальные значения

$$\eta_1 = \eta_{\text{Hansen}} + 0{,}1 \quad \text{и} \quad \eta_2 = \eta_{\text{Hansen}}$$

получаются из так называемого приближения Хансена (Hansen):

$$\eta_{\text{Hansen}} = \frac{12}{22} + \frac{10}{22}\sqrt{1 + \frac{44}{9}\,\frac{m}{l + 5/6}}. \tag{9.8}$$

Чтобы запрограммировать эти выражения, нужно сначала определить типы данных

```
    TYPE INDEX= (X,Y,Z);
         VECTOR= ARRAY[INDEX] OF REAL;
```

и функции

```
(*------------------------------------------------------------*)
(* DOT: Скалярное произведение двух векторов                  *)
(*------------------------------------------------------------*)
FUNCTION DOT(A,B:VECTOR):REAL;
  BEGIN
   DOT:=A[X]*B[X]+A[Y]*B[Y]+A[Z]*B[Z];
  END;

(*------------------------------------------------------------*)
(* NORM: Длина вектора                                        *)
(*------------------------------------------------------------*)
  FUNCTION NORM(A:VECTOR):REAL;
  BEGIN
   NORM:= SQRT(DOT(A,A));
  END;
(*------------------------------------------------------------*)
```

Таким образом, для вычисления отношения сектор—треугольник получится следующая подпрограмма.

```
(*------------------------------------------------------------*)
(* FIND_ETA: Определение отношения сектор-треугольник         *)
(*           по двум положениям и разности моментов           *)
(*------------------------------------------------------------*)
FUNCTION FIND_ETA ( RA,RB: VECTOR; TAU: REAL ): REAL;

  CONST DELTA = 1.0E-9;   MAXIT = 30;

  VAR   KAPPA,M,L,SA,SB,ETA_MIN,ETA1,ETA2,F1,F2,D_ETA: REAL;
          I: INTEGER;

  (* F(eta) = 1 - eta + (m/eta**2)*W(m/eta**2-l) *)
  FUNCTION F ( ETA,M,L: REAL ): REAL;
    CONST EPS =1.0E-10;
    VAR W,WW,A,S,N,G,E: REAL;
  BEGIN
    W:= M/(ETA*ETA)-L;
    IF  ( ABS(W)<0.1)
         THEN (* разложение в ряд                             *)
           BEGIN
             A:=4.0/3.0;  WW:=A;  N:=0.0;
             REPEAT
             N:=N+1;        A :=A*W*(N+2.0)/(N+1.5); WW:=WW+A;
             UNTIL ABS(A)<EPS;
```

```
          END
        ELSE
          IF  (W>0)
             THEN (* W=(2g-sin2g)/(sin(g)**3),                    *)
                  (*  g=2*arcsin(sqrt(w))                         *)
               BEGIN
                G :=  2.0*ARCTAN(SQRT(W/(1.0-W)));    S:=SIN(G);
                WW:=(2.0*G-SIN(2.0*G))/(S*S*S);
               END
    ELSE (* W=(sh2g-2g)/(sh(g)**3), g=2*arsh(sqrt(-w))            *)
             BEGIN
               G :=  2.0*LN(SQRT(-W)+SQRT(1.0-W));
               E :=  EXP(G);   S:=0.5*(E-1.0/E);   E:=E*E;
               WW:=(0.5*(E-1 0/E)-2.0*G)/(S*S*S);
  END;
        F := 1 0-ETA+(W+L)*WW;
     END; (* FIND_ETA.F *)

  BEGIN

    SA      := NORM(RA); SB := NORM(RB);
    KAPPA := SQRT(2.0*(SA*SB+DOT(RA,RB))),
    M       := TAU*TAU / (KAPPA*KAPPA*KAPPA);
    L        := (SA+SB)/(2.0*KAPPA)  -  0.5;
    ETA_MIN:= SQRT(M/(L+1.0));

    (* начальное значение: приближение Хансена                    *)
    ETA2:= ( 12.0 + 10.0*SQRT(1 0+(44.0/9.0)*M/(L
                                          +5.0/6 0)) ) / 22.0;
    ETA1:= ETA2 + 0.1;  F1:=  F(ETA1,M,L);
                        F2:=  F(ETA2,M,L);    I:=0;

    (* метод секущих                                              *)
    WHILE ( (ABS(F2-F1)>DELTA) AND (I<MAXIT) ) DO
        BEGIN
        D_ETA:=  -F2*(ETA2-ETA1)/(F2-F1);  ETA1:=ETA2;  F1:=F2;
        WHILE    (ETA2+D_ETA<=ETA_MIN) DO D_ETA:=0.5*D_ETA;
        ETA2  :=  ETA2+D_ETA;  F2:=F(ETA2,M,L);
      END;
   IF (I=MAXIT) THEN  WRITELN('проблема сходимости в  FIND_ETA');
   FIND_ETA := ETA2;
  END;
(*-------------------------------------------------------------*)
```

Так как язык Паскаль не имеет гиперболических функций, их нужно заменить экспоненциальными и логарифмическими функциями:

$$\text{sh } x = \frac{1}{2} (\exp x - 1/\exp x) ,$$

$$\text{sh } (2x) = \frac{1}{2} ((\exp x)^2 - 1/(\exp x)^2) ,$$

$$\text{arcsh } x = \ln\left[x + \sqrt{1 + x^2} \right] .$$

Аналогично можно использовать

$$\arcsin (x) = \text{arctg} \frac{x}{\sqrt{1 - x^2}} .$$

9.1.2. Элементы орбиты

Орбита небесного тела, проходящая через точки $\mathbf{r}_a$ и $\mathbf{r}_b$, никогда не выходит из плоскости, проведенной через эти две точки и Солнце. Для определения наклона i этой плоскости к эклиптике и вычисления долготы восходящего узла вводят векторы единичной длины $\mathbf{e}_a$ и $\mathbf{e}_0$, лежащие в плоскости орбиты:

$$\mathbf{e}_a = \frac{\mathbf{r}_a}{|\mathbf{r}_a|} , \tag{9 9}$$

$$\mathbf{e}_0 = \frac{\mathbf{r}_0}{|\mathbf{r}_0|} , \text{ где } \mathbf{r}_0 = \mathbf{r}_b - (\mathbf{r}_b \cdot \mathbf{e}_a) \, \mathbf{e}_a . \tag{9.10}$$

Смысл этих векторов показан на рис.9.2. В то время как вектор $\mathbf{e}_a$ указывает в направлении $\mathbf{r}_a$, $\mathbf{r}_0$ и $\mathbf{e}_0$ перпендикулярны к нему. Векторное произведение $\mathbf{e}_a$ и $\mathbf{e}_0$ даст гауссов вектор $\mathbf{R}$, перпендикулярный

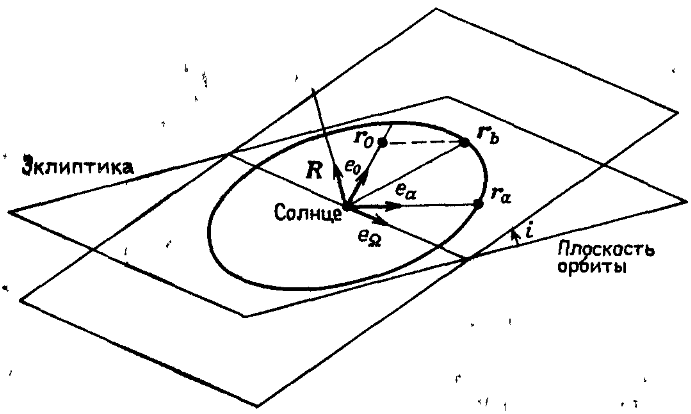


Рис.9.2. Вспомогательные величины для **определения положения плоскости** орбиты.

плоскости орбиты и также нормированный на единичную длину ($|\mathbf{R}| = 1$)

$$\mathbf{R} = \mathbf{e}_a \times \mathbf{e}_0 , \quad \begin{bmatrix} R_x \\ R_y \\ R_z \end{bmatrix} = \begin{bmatrix} y_a z_0 - z_a y_0 \\ z_a x_0 - x_a z_0 \\ x_a y_0 - y_a x_0 \end{bmatrix} . \tag{9.11}$$

Вектор $\mathbf{R}$ направлен к точке с эклиптической долготой $l = \Omega - 90^\circ$ и широтой $b = 90^\circ - i$ и поэтому может быть выражен через элементы Ω и i:

$$\mathbf{R} = \begin{bmatrix} R_x \\ R_y \\ R_z \end{bmatrix} = \begin{bmatrix} +\cos (90^\circ - i) \cos (\Omega - 90^\circ) \\ +\cos (90^\circ - i) \sin (\Omega - 90^\circ) \\ +\sin (90^\circ - i) \end{bmatrix} = \begin{bmatrix} -\sin i \sin \Omega \\ -\sin i \cos \Omega \\ +\cos i \end{bmatrix}. \tag{9.12}$$

Таким образом получают три уравнения, из которых однозначно определяются долгота узла и наклон орбиты.

$$\Omega = 90^\circ + \operatorname{arctg} (R_y/R_x) = \operatorname{arctg} (-R_x/R_y) , \tag{9.13}$$

$$i = 90^\circ - \arcsin (R_z) \tag{9.14}$$

Значения обоих углов относятся к тому же равноденствию, что и векторы $\mathbf{r}_a$ и $\mathbf{r}_b$ Исходя из положения линии узлов можно вычислить аргумент широты u_a, который является углом между вектором $\mathbf{r}_a$ и направлением на восходящий узел Для этого угла

$$\cos u_a = \mathbf{e}_a \cdot \mathbf{e}_\Omega = x_a \cdot \cos \Omega + y_a \cdot \sin \Omega ,$$

$$\cos (u_a + 90^\circ) = \mathbf{e}_0 \cdot \mathbf{e}_\Omega = x_0 \cdot \cos \Omega + y_0 \cdot \sin \Omega ,$$

где

$$\mathbf{e}_\Omega = \begin{bmatrix} \cos \Omega \\ \sin \Omega \\ 0 \end{bmatrix}$$

— вектор единичной длины в направлении линии узлов. Поэтому

$$u_a = \operatorname{arctg} \left[\frac{-x_0 \cdot \cos \Omega - y_0 \cdot \sin \Omega}{+x_a \cdot \cos \Omega + y_a \cdot \sin \Omega} \right] =$$

$$= \operatorname{arctg} \left[\frac{+x_0 R_y - y_0 R_x}{-x_a R_y + y_a R_x} \right] . \tag{9.15}$$

Для вычисления остальных элементов орбиты необходимо знать отношение сектор—треугольник, которое мы вычислим способом, изложенным в предыдущем разделе. С его помощью затем определяется параметр орбиты

$$p = \left[\frac{2\,\Delta\,\eta}{\tau} \right]^2$$

через площадь, ограниченную векторами $\mathbf{r}_a$ и $\mathbf{r}_b$

$$\Delta = \frac{1}{2}\, r_a r_b \sin(\nu_b - \nu_a) = \frac{1}{2}\, r_a r_0 ,$$

и промежуток времени τ.

Форма орбиты описывается эксцентриситетом e, определяемым уравнением конического сечения

$$r = \frac{p}{1 + e \cdot \cos\nu} .$$

Решив его относительно $e\cos\nu$, получим·

$$e \cdot \cos\nu_a = p/r_a - 1,$$

$$e \cdot \cos\nu_b = p/r_b - 1$$

Если мы примем во внимание, что

$$\cos\nu_b = \cos\nu_a \cos(\nu_b - \nu_a) - \sin\nu_a \sin(\nu_b - \nu_a) =$$

$$= \cos\nu_a \cdot \left[\frac{\mathbf{r}_b\,\mathbf{e}_a}{r_b} \right] - \sin\nu_a \cdot \left[\frac{r_0}{r_b} \right] ,$$

получим окончательно два уравнения:

$$e\cdot\cos\nu_a = p/r_a - 1 ,$$

$$e\cdot\sin\nu_a = \left\{ p/r_a - 1) \left[\frac{\mathbf{r}_b \cdot \mathbf{e}_a}{r_b} \right] - (p/r_b - 1) \right\} / \left[\frac{r_0}{r_b} \right] ,$$

из которых найдем эксцентриситет и истинную аномалию в момент t_a:

$$e = \sqrt{(e\cos(\nu_a))^2 + (e\sin(\nu_a))^2} ,$$

$$\nu_a = \operatorname{arctg} \left[\frac{e \cdot \sin(\nu_a)}{e \cdot \cos(\nu_a)} \right] .$$

Из разности аргумента широты и истинной аномалии получают аргумент и долготу перигелия:

$$\omega = u_a - \nu_a , \tag{9.16}$$

$$\varpi = u_a - \nu_a + \Omega \tag{9.17}$$

Эти величины позволяют установить, является ли орбита, соединяющая $\mathbf{r}_a$ и $\mathbf{r}_b$, эллипсом $(e < 1)$ или гиперболой $(e > 1)$. Параболические орбиты мы не рассматриваем, так как на практике при определении орбиты вероятность того, что эксцентриситет e будет равен точно единице, очень мала Параметр орбиты и эксцентриситет позволяют получить также большую полуось орбиты и расстояние до перигелия

$$a = \frac{p}{1 - e^2}, \tag{9.18}$$

$$q = \frac{p}{1 + e}. \tag{9.19}$$

Следует учитывать, что по определению большая полуось гиперболической орбиты отрицательна.

Таким образом, уже определены элементы орбиты, задающие ее форму (e), размер (a) и пространственное положение (i, Ω, ω). Последним, шестым элементом орбиты, подлежащим определению, является момент прохождения перигелия, указывающий, когда тело проходит ближайшую к Солнцу точку. Для эллиптической орбиты в первую очередь определяют эксцентрическую аномалию E_a по формулам

$$\cos E_a = \frac{\cos \nu_a + e}{1 + e \cdot \cos \nu_a},$$

$$\sin E_a = \frac{\sqrt{1 - e^2}\, \sin \nu_a}{1 + e \cdot \cos \nu_a}$$

Они получаются, если из (4.5) с помощью уравнения конического сечения исключить радиус, а затем решить его относительно эксцентрической аномалии Значение средней аномалии M, соответствующее E, получается из уравнения Кеплера.

$$M_a = E_a - e \cdot \sin E_a \text{ (радиан)}.$$

При орбитальном периоде

$$T = 2\pi \sqrt{\frac{a^3}{GM_\odot}},$$

который получается из третьего закона Кеплера, средняя аномалия за сутки меняется на

$$n = 2\pi \frac{1^d}{T} = \sqrt{\frac{GM_\odot}{a^3}} \cdot 1^d,$$

так что момент прохождения через перигелий будет определяться соотношением

$$t_0 = t_a - M_a / \sqrt{GM_\odot / a^3}. \tag{9.20}$$

Для гиперболических орбит соответствующие соотношения будут выглядеть следующим образом:

$$\sin H_a = \frac{\sqrt{e^2 - 1}\,\sin \nu_a}{1 + e \cos \nu_a}, \tag{9.21}$$

$$M_a = e \sin H_a - H_a, \tag{9.22}$$

$$t_0 = t_a - M_a / \sqrt{GM_{\odot}/|a|^3}. \tag{9.23}$$

Полное определение орбитальных элементов по двум точкам орбиты осуществляется подпрограммой ELEMENT. Для упрощения ввода $\mathbf{r}_a$ и $\mathbf{r}_b$ задаются как массивы описанного выше типа VECTOR.

```
(*-----------------------------------------------------------------------*)
(* CROSS: Векторное произведение двух векторов                           *)
(*-----------------------------------------------------------------------*)
PROCEDURE CROSS(A,B:VECTOR;VAR C:VECTOR);
  BEGIN
    C[X]:=A[Y]*B[Z]-A[Z]*B[Y];
    C[Y]:=A[Z]*B[X]-A[X]*B[Z];
    C[Z]:=A[X]*B[Y]-A[Y]*B[X];
  END;
(*-----------------------------------------------------------------------*)
(* ELEMENT: Вычисление орбит.элементов по двум точкам орбиты             *)
(*          для эллиптических и гиперболических орбит                    *)
(* JDA,JDB: моменты прохождения точек A и B (юлианские даты)             *)
(* RA, RB : векторы положения точек орбиты A и B                         *)
(* TP     : момент прохождения перигелия (в юл. столет. с J2000)         *)
(* Q      : расстояние до перигелия                                      *)
(* ECC    : эксцентриситет                                               *)
(* INC    : наклон орбиты (град.)                                        *)
(* LAN    : долгота восходящего узла (град.)                             *)
(* AOP    : аргумент перигелия (град.)                                   *)
(*-----------------------------------------------------------------------*)
PROCEDURE ELEMENT ( JDA,JDB: REAL; RA,RB: VECTOR;
                    VAR TP, Q ECC,INC,LAN,AOP: REAL);
  CONST KGAUSS=0.01720209895;
        RAD   =0.01745329252; (*  180/pi  *)

  VAR   TAU,ETA,P,AX,N,NY,E,M,U             : REAL;
        SA,SB,S0,FAC,DUMMY,SHH               : REAL;
        COS_DNY,SIN_DNY,ECOS_NY,ESIN_NY:REAL;
        EA,R0,E0,R                           : VECTOR;
        I                                    : INDEX;

  BEGIN

 (* вычисление вектора R0 (составляющая RB, перпендикулярная RA)         *)
```

```
(*                           и величин RA, RB и R0                          *)
   SA :=NORM(RA);     FOR I:=X TO Z DO EA[I]:=RA[I]/SA;
   SB :=NORM(RB);
   FAC:=DOT(RB,EA); FOR I:=X TO Z DOR0[I]:=RB[I]-FAC*EA[I];
   S0 :=NORM(R0);     FOR I:=X TO Z DOE0[I]:=R0[I]/S0;

   (* наклон орбиты и восходящий узел *)
   CROSS (EA,E0,R);
   POLAR ( -R[Y],R[X],R[Z], DUMMY,INC,LAN );   INC := 90.0-INC;
   U:=ATN2 ( (+E0[ X]*R[Y]-E0[Y]*R[X]) ,
                (-EA[X]*R[Y]+EA[Y]*R[X])      );
   IF INC=0.0 THEN U:=ATN2(RA[Y],RA[X]);

   (* параметр орбиты p                                                      *)
   TAU:=KGAUSS * ABS(JDB-JDA);ETA:=FIND_ETA(RA,RB,TAU);
   P  :=SA*S0*ETA / TAU;         P  :=P*P;

   (* эксцентриситет, истинная аномалия и долгота перигелия                  *)
   COS_DNY:=FAC/SB;  SIN_DNY:=S0/SB;
   ECOS_NY:=P/SA-1.0; ESIN_NY:=(ECOS_NY*COS_DNY-(P/SB
                                              -1.0)) / SIN_DNY;
   POLAR  ( ECOS_NY,ESIN_NY,0.0, ECC,DUMMY,NY );
   AOP    := U-NY;  WHILE (AOP<0.0) DO AOP:=AOP+360.0;

   (* расстояние до перигелия, большая полуось и среднее                     *)
   (* суточное движение                                                      *)
   Q:= P/(1.0+ECC);  AX:= Q/(1.0-ECC);
   N:= KGAUSS / SQRT(ABS(AX*AX*AX));
   (* средняя аномалия и момент прохождения перигелия                        *)
   IF  (ECC<1.0)
     THEN
       BEGIN
        E:= ATN2 ( SQRT((1.0-ECC)*(1.0+ECC))*ESIN_NY,
                           ECOS_NY+ECC*ECC );
        E:= RAD*E; M:=E-ECC*SIN(E);
     END
  ELSE
  BEGIN
    SHH:= SQRT((ECC-1.0)*(ECC+1.0))*ESIN_NY / (ECC
                                          +ECC*ECOS_NY);
    M  := ECC*SHH - LN(SHH+SQRT(1.0+SHH*SHH))
       END;
    TP := ( (JDA-M/N) - 2451545 0 ) / 36525.0;

 END;
(*-----------------------------------------------------------------------*)
```

9.2. Геометрия геоцентрических наблюдений

Если вектор $\mathbf{r}$ представляет гелиоцентрическое положение планеты (относительно Солнца), а $\mathbf{R}$—геоцентрическое положение Солнца (относительно Земли), то геоцентрическое положение планеты будет определяться как

$$\rho \mathbf{e} = \mathbf{R} + \mathbf{r} . \qquad (9.24)$$

Здесь ρ—расстояние планеты от Земли, а $\mathbf{e}$—вектор единичной длины, направленный от Земли к планете. В эклиптических или экваториальных координатах $\mathbf{e}$ имеет компоненты:

$$\begin{bmatrix} \cos\lambda\cos\beta \\ \sin\lambda\cos\beta \\ \sin\beta \end{bmatrix} \quad \text{или} \quad \begin{bmatrix} \cos\alpha\cos\delta \\ \sin\alpha\cos\delta \\ \sin\delta \end{bmatrix},$$

где λ и β—соответственно эклиптические долгота и широта, а α и δ—прямое восхождение и склонение планеты, наблюдаемой с Земли.

Измерив координаты планеты на видимой небесной сфере (например, α и δ), можно определить наблюдаемое направление и тем самым вектор $\mathbf{e}$. Расстояние ρ неизвестно и потому должно быть вычислено в процессе определения орбиты. Для однозначного определения орбиты нужно иметь по меньшей мере три наблюдения $\mathbf{e}_1$, $\mathbf{e}_2$ и $\mathbf{e}_3$. Предположим, что координаты Солнца $\mathbf{R}_1$, $\mathbf{R}_2$ и $\mathbf{R}_3$ в моменты наблюдений известны. Тогда, исходя из этих данных, можно попытаться вычислить расстояния ρ_1, ρ_2 и ρ_3. Только когда эти расстояния известны, можно определить гелиоцентрические положения, которые задают орбиту и позволяют вычислить ее элементы. Геометрические зависимости, получаемые из набора трех положений планеты, нужно теперь представить в форме, удобной для определения орбиты.

В моменты $t_1 < t_2 < t_3$ планета относительно Солнца находится в положениях $\mathbf{r}_1$, $\mathbf{r}_2$ и $\mathbf{r}_3$. При невозмущенном кеплеровском движении все эти точки вместе с Солнцем лежат в одной плоскости, поэтому один вектор положения всегда можно представить в виде удобной комбинации двух других. Выбрав для этого $\mathbf{r}_2$, можно написать:

$$\mathbf{r}_2 = n_1\mathbf{r}_1 + n_3\mathbf{r}_3 \quad \text{(уравнение орбитальной плоскости).} \qquad (9.25)$$

Коэффициенты n_1 и n_3 зависят от взаимного расположения векторов $\mathbf{r}_1$, $\mathbf{r}_2$ и $\mathbf{r}_3$. При принятом далее предположении, что рассматриваемая полная дуга орбиты меньше 180°, оба коэффициента положительны. Сопоставив (9.24) и (9.25), получим

$$(\rho_2\mathbf{e}_2 - \mathbf{R}_2) = n_1(\rho_1\mathbf{e}_1 - \mathbf{R}_1) + n_3(\rho_3\mathbf{e}_3 - \mathbf{R}_3)$$

или, сделав перестановку,

$$n_1\rho_1\mathbf{e}_1 - \rho_2\mathbf{e}_2 + n_3\rho_3\mathbf{e}_3 = n_1\mathbf{R}_1 - \mathbf{R}_2 + n_3\mathbf{R}_3 . \tag{9.26}$$

Определим теперь векторы

$$\mathbf{d}_1 = \mathbf{e}_2\times\mathbf{e}_3, \qquad \mathbf{d}_2 = \mathbf{e}_3\times\mathbf{e}_1, \qquad \mathbf{d}_3 = \mathbf{e}_1\times\mathbf{e}_2 ,$$

тогда из-за свойства векторного произведения вектор $\mathbf{d}_1$ будет перпендикулярен $\mathbf{e}_2$ и $\mathbf{e}_3$, $\mathbf{d}_2$ — перпендикулярен $\mathbf{e}_3$ и $\mathbf{e}_1$, а $\mathbf{d}_3$ — перпендикулярен $\mathbf{e}_1$ и $\mathbf{e}_2$. Поэтому скалярное произведение $\mathbf{e}_i\cdot\mathbf{d}_j$ отличается от нуля только при $i=j$. Умножив (9.26) на $\mathbf{d}_1$, $\mathbf{d}_2$ и $\mathbf{d}_3$, получим

$$n_1\rho_1(\mathbf{e}_1\cdot\mathbf{d}_1) = (n_1\mathbf{R}_1 - \mathbf{R}_2 + n_3\mathbf{R}_3)\cdot\mathbf{d}_1,$$

$$-\rho_2(\mathbf{e}_2\cdot\mathbf{d}_2) = (n_1\mathbf{R}_1 - \mathbf{R}_2 + n_3\mathbf{R}_3)\cdot\mathbf{d}_2,$$

$$n_3\rho_3(\mathbf{e}_3\cdot\mathbf{d}_3) = (n_1\mathbf{R}_1 - \mathbf{R}_2 + n_3\mathbf{R}_3)\cdot\mathbf{d}_3 .$$

Обозначив

$$D = \mathbf{e}_1\cdot(\mathbf{e}_2\times\mathbf{e}_3) = \mathbf{e}_2\cdot(\mathbf{e}_3\times\mathbf{e}_1) = \mathbf{e}_3\cdot(\mathbf{e}_1\times\mathbf{e}_2) =$$

$$= \mathbf{e}_1\cdot\mathbf{d}_1 = \mathbf{e}_2\cdot\mathbf{d}_2 = \mathbf{e}_3\cdot\mathbf{d}_3$$

и

$$D_{ij} = \mathbf{d}_i\cdot\mathbf{R}_j,$$

получим три уравнения:

$$\rho_1 = \frac{1}{n_1 D}(n_1 D_{11} - D_{12} + n_3 D_{13}),$$

$$\rho_2 = \frac{1}{-D}(n_1 D_{21} - D_{22} + n_3 D_{23}), \tag{9.27}$$

$$\rho_3 = \frac{1}{n_3 D}(n_1 D_{31} - D_{32} + n_3 D_{33}),$$

представляющие ρ_1, ρ_2 и ρ_3 через n_1 и n_3 и векторы $\mathbf{e}_{1\ldots3}$ и $\mathbf{R}_{1\ldots3}$. Поначалу кажется, что это преобразование не принесло нам большой выгоды, так как n_1 и n_3 неизвестны. Однако важно, что используя уравнение орбитальной плоскости, мы уменьшили число неизвестных от трех ($\rho_{1\ldots3}$) до двух ($n_{1,3}$). Введенным нами коэффициентам большое значение придает то обстоятельство, что они, как будет показано, в очень хорошем приближении выражаются через отношение известных нам временных интервалов.

Рассмотрим вновь уравнение орбитальной плоскости (9.25). Умножив векторно обе части на $\mathbf{r}_3$ или $\mathbf{r}_1$ и учтя, что векторное произведение вектора самого на себя равно нулю, получим

$$(\mathbf{r}_2 \times \mathbf{r}_3) = n_1 (\mathbf{r}_1 \times \mathbf{r}_3), \qquad (\mathbf{r}_1 \times \mathbf{r}_2) = n_3 (\mathbf{r}_1 \times \mathbf{r}_3)$$

и

$$n_1 = \frac{| \mathbf{r}_2 \times \mathbf{r}_3 |}{| \mathbf{r}_1 \times \mathbf{r}_3 |}, \qquad n_3 = \frac{| \mathbf{r}_1 \times \mathbf{r}_2 |}{| \mathbf{r}_1 \times \mathbf{r}_3 |}.$$

Как известно, площадь Δ треугольника, построенного на векторах $\mathbf{r}_a$ и $\mathbf{r}_b$, равна

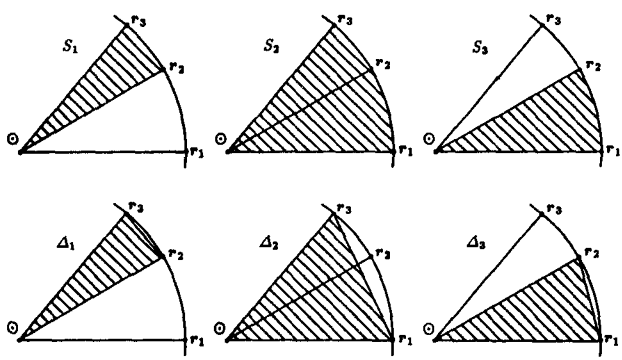
$$\Delta = \frac{1}{2} | \mathbf{r}_a \times \mathbf{r}_b | .$$

Поэтому n_1 и n_3 можно интерпретировать как отношения площадей треугольников, построенных на $\mathbf{r}_1$, $\mathbf{r}_2$ и $\mathbf{r}_3$ (рис.9.3):

$$n_1 = \frac{\Delta_1}{\Delta_2}, \qquad n_3 = \frac{\Delta_3}{\Delta_2}.$$

Площади малых треугольников (особенно при малых орбитальных дугах) мало отличаются от площадей соответствующих секторов $S_i = \eta_i \Delta_i$, пропорциональных времени τ_i:

$$n_1 = \frac{\eta_2}{\eta_1} \cdot \frac{\tau_1}{\tau_2} \approx \frac{\tau_1}{\tau_2}, \qquad n_3 = \frac{\eta_2}{\eta_3} \cdot \frac{\tau_3}{\tau_2} \approx \frac{\tau_3}{\tau_2}.$$



$\tau_1 = \sqrt{GM_\odot}(t_3 - t_2)$ $\qquad$ $\tau_2 = \sqrt{GM_\odot}(t_3 - t_1)$ $\qquad$ $\tau_3 = \sqrt{GM_\odot}(t_2 - t_1)$

Рис. 9.3. Площади сектора и треугольника и интервалы времени для трех гелиоцентрических положений.

Таким образом, нам уже известны приближенные значения n_1 и n_3, которые посредством (9.27) позволяют в первом приближении определить геоцентрические расстояния (ρ_1, ρ_2, ρ_3).

9.3. Метод уточнения

Приведенные до сих пор соображения позволяют утверждать, что орбита небесного тела может быть однозначно определена, если известны его положения в два момента времени. То же самое касается элементов орбиты и отношения площадей сектора и треугольника. С другой стороны, если известно отношение сектор—треугольник для ряда из трех наблюдаемых положений, то можно вычислить геоцентрические и гелиоцентрические векторы положения. Упрощенный метод Гаусса определения орбит, описываемый нами, основывается как раз на этом обстоятельстве.

Предположим, что даны три геоцентрических направления наблюдения ($\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{e}_3$), соответствующие геоцентрические координаты Солнца ($\mathbf{R}_1$, $\mathbf{R}_2$, $\mathbf{R}_3$) и моменты (τ_1, τ_2, τ_3). С использованием этих исходных данных следует провести следующие вычисления.

1. Принять $n_1 = \tau_1/\tau_2$ и $n_3 = \tau_3/\tau_2$ как исходные приближения для отношения площадей треугольников.
2. Повторять шаги (a)—(d) до тех пор, пока n_1, n_3 и остальные величины не перестанут существенно меняться.

(a) Вычислить геоцентрические расстояния (ρ_1, ρ_2, ρ_3) по (9.27).

(b) Вычислить векторы гелиоцентрических положений ($\mathbf{r}_1$, $\mathbf{r}_2$, $\mathbf{r}_3$) при помощи соотношения (9.24).

(c) Вычислить отношения сектор—треугольник (η_1, η_2, η_3) для каждой пары векторов гелиоцентрических положений и соответствующих интервалов времени по (9.5).

(d) Вычислить уточненные значения $n_1 = (\eta_2/\eta_1)\ (\tau_1/\tau_2)$ и $n_3 = (\eta_2/\eta_3)\ (\tau_3/\tau_2)$ для отношения площадей треугольников.

3. Вычислить элементы орбиты по последним значениям $\mathbf{r}_1$, $\mathbf{r}_3$ и τ_2.

Определенная таким образом орбита имеет требуемое свойство: любое движущееся по ней тело в моменты t_1, t_2 и t_3 будет находится соответственно в точках $\mathbf{r}_1$, $\mathbf{r}_2$ и $\mathbf{r}_3$, лежащих относительно Земли в направлениях $\mathbf{e}_1$, $\mathbf{e}_2$ и $\mathbf{e}_3$.

Таким образом, в принципе задача определения орбиты нами решена. Для практического применения метода нужно лишь небольшое изменение схемы вычислений. До сих пор задача определения орбиты рассматривалась нами так, что в каждый момент времени наблюдаемое положение планеты или кометы отождествлялось с ее геометрическим положением на этот момент. В действительности между испусканием кванта света и его наблюдением проходит некоторое время, которое определяется конечной скоростью света

$$c = 173{,}1 \text{ а.е./сут},$$

$$1/c = 0{,}05776 \text{ сут/а.е.}$$

Следовательно, векторы $\mathbf{e}_i$, которые задают наблюдаемое направление, направлены от гелиоцентрических положений $-\mathbf{R}_i$, в которых находится Земля в моменты наблюдений, к трем точкам $\mathbf{r}_i = \rho_i\mathbf{e}_i + \mathbf{R}_i$, в которых находилась планета в более ранние моменты времени $t_i < t'_i$ (рис 9.4). Время, прошедшее между испусканием света и наблюдением, $t'_i - t_i$, равно при этом времени, которое требуется свету для того, чтобы пройти отрезок ρ_i:

$$\rho_i = c\,(t'_i - t_i).$$

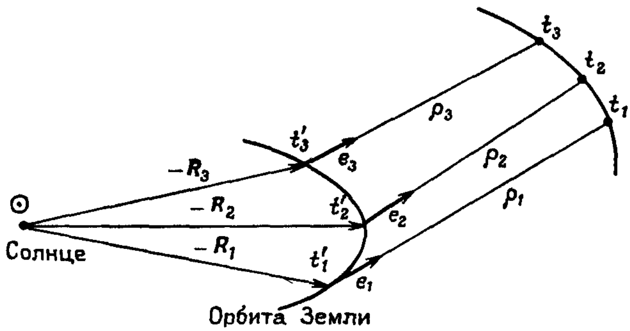


Рис. 9.4. Учет времени распространения света.

Здесь поначалу возникает затруднение, поскольку при вычислении промежутка времени τ_i не известен момент испускания света. Однако в ходе определения орбиты получаются все более точные геоцентрические расстояния (ρ_1, ρ_2, ρ_3), так что учет времени распространения света не вызывает затруднений. Моменты наблюдений нужны лишь при определении начального приближения

$$n_1 = \frac{\tau_1}{\tau_2} \approx \frac{t'_3 - t'_2}{t'_3 - t'_1}, \qquad n_1 = \frac{\tau_3}{\tau_2} \approx \frac{t'_2 - t'_1}{t'_3 - t'_1}$$

отношения площадей треугольников. Далее при каждом шаге итерации промежутки времени вычисляются заново, пока не станут известными действительные геоцентрические расстояния:

$$\tau_1 = \sqrt{GM_\odot}\,(t_3 - t_2), \quad \tau_2 = \sqrt{GM_\odot}\,(t_3 - t_1), \quad \tau_3 = \sqrt{GM_\odot}\,(t_2 - t_1)$$

при

$$t_i = t'_i - \rho_i \cdot 0{,}05776 \text{ сут/а.е.} \qquad (i = 1...3)\ .$$

Вспомогательные геометрические величины $\mathbf{d}_i$, D_{ij} и D зависят только от векторов направления $\mathbf{e}_i$ и координат Солнца $\mathbf{R}_i$ и поэтому не требуют поправок

9.4. Многозначные решения

Упрощенный метод Гаусса определения орбит всегда (кроме исключительных ситуаций, когда итерации не сходятся) дает результат, удовлетворяющий всем уравнениям задачи определения орбиты. В сомнительных случаях этот результат подтверждается соответствующим проверочным вычислением трех наблюдений с найденными элементами орбиты. При этом, однако, не исключается получение решений, представляющихся очевидно неправильными или бессмысленными. Особенно часто это происходит в случае гиперболических орбит с большим эксцентриситетом, которые редко встречаются на практике[1] Причина состоит в том, что определение орбиты по трем наблюдениям не всегда однозначно. Это особенно хорошо видно в случае решения, соответствующего орбите Земли Наблюдатель сам движется по почти невозмущенной кеплеровской орбите вокруг Солнца и, кроме того, всегда расположен на заданном направлении наблюдения. Решение, относящееся к орбите Земли, легко узнать по полуоси ($a \approx 1$), эксцентриситету ($e \approx 0$) и наклону к эклиптике ($i \approx 0°$)

Общий анализ неоднозначностей в определении орбит обычно ограничивают малыми дугами орбиты Если при таком условии рассматривать взаимное расположение Солнца, Земли и наблюдаемого небесного тела, то появляются четыре различных области, разделенные орбитой Земли и так называемой *граничной линией Шарлье* (рис.9.5). Например, в случае астероида, наблюдаемого вблизи противостояния, решение получается однозначным, если не считать упоминавшегося выше решения, относящегося к земной орбите. С другой стороны, если комета во время наблюдений находится внутри зоны каплеобразной формы, окружающей Солнце, или в области соединения вне орбиты Земли, то в принципе возможны два решения задачи определения орбиты.

Представленный здесь упрощенный вариант определения орбиты не дает возможности получить все эти решения и различить их. Здесь необходим полный метод Гаусса для определения орбиты, описанный в литературе.

9.5. Программа ORBDET

Используя представленную здесь программу ORBDET, можно по трем наблюдениям планеты или астероида определить элементы орбиты. Вначале по заданным значениям прямого восхождения и склонения определяют векторы направления наблюдений единичной длины, а также коор-

---

[1] Все без исключения известные кометы имеют эксцентриситет менее $e = 1{,}1$.

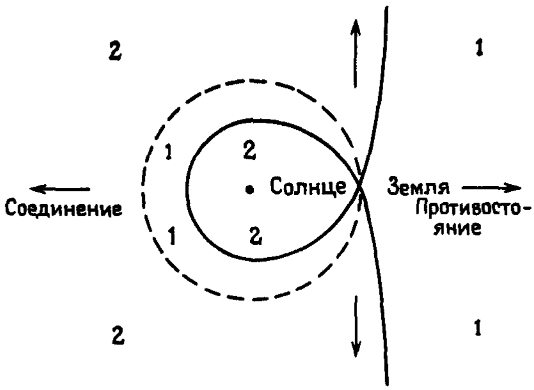


Рис.9.5. Граничная линия, разделяющая области однозначного и неоднозначного определения орбиты.

динаты Солнца (START). Так как искомые элементы орбиты должны быть отнесены к эклиптике, все вычисления проводятся в одной эклиптической системе координат, равноденствие которой можно выбирать. Само определение орбиты по упрощенному методу Гаусса обеспечивается процедурой GAUSS. Две подпрограммы (DUMPELEM, SAVEELEM) выводят элементы орбиты на экран и в файл с фиксированным именем ORBOUT(.DAT). Формат этого файла соответствует формату ввода программы COMET гл.4, так что последующее вычисление эфемерид не представляет трудностей.

Из-за большого объема вводимых данных они объединены в файл под именем ORBINP(DAT) Файл начинается со строки комментария, что облегчает идентификацию данных Три следующие строки содержат даты (день, месяц, год и час с десятичными знаками), экваториальные координаты (прямое восхождение в часах, минутах и секундах с десятичными знаками и склонение в градусах, минутах и секундах с десятичными знаками) и примечания. Следующие две строки содержат равноденствие, к которому отнесены наблюдаемые положения, и желаемое равноденствие вычисляемых элементов орбиты. Приведенные здесь в качестве примера данные содержат несколько наблюдений астероида Церера, которые Гаусс использовал для иллюстрации своего метода определения орбиты.

```
Церера (пример Гаусса)
05 09 1805  24 165   6 23 57.54  22 21 27.08 !  три наблюдения
17 01 1806  22.095   6 45 14.69  30 21 24.20 !  формат: дата (дмгч)
```

```
23 05 1806   20.399   8 07 44.60  28 02 47.04  ! RA (чмс), Dec (гмс)
1806.0                                         ! равноденствие
1806.0                                         ! желаемое равноденствие
```

При определении орбиты считается, что все наблюдения заданы в форме астрометрических координат, освобожденных от влияния звездной аберрации. В частности, так всегда бывает, когда координаты объекта получены путем сравнения с координатами окрестных звезд в звездном каталоге. Примером такого подхода может служить измерение пластинки звездного поля, которое чаще всего выполняется для определения движения кометы или астероида.

```
(*-----------------------------------------------------------*)
(*                          ORBDET                           *)
(*          Определение орбиты по трем наблюдениям           *)
(*            Метод Гаусса, упрощенный Буцериусом            *)
(*                     Версия 12.02.89                       *)
(*-----------------------------------------------------------*)

PROGRAM ORBDET(INPUT,OUTPUT,ORBINP,ORBOUT);
  TYPE INDEX = (X,Y,Z);
       VECTOR= ARRAY[INDEX] OF REAL;
       REAL3  = ARRAY[1.. 3] OF REAL;
       REAL33 = ARRAY[1.. 3] OF REAL3;
       MAT3X  = ARRAY[1.. 3] OF VECTOR;
       CHAR80= ARRAY[1 .80] OF CHAR,

  VAR  TEQX                        : REAL;
       TP,Q,ECC,INC,LAN,AOP.       REAL;
       JD0                         : REAL3;
       RSUN,E                      : MAT3X;
       HEADER                      : CHAR80;
       ORBINP,ORBOUT               : TEXT;
(*-----------------------------------------------------------*)
(* Сюда вводятся следующие подпрограммы в указанной          *)
(* последовательности:                                       *)
(* SN,CS,TN,ATN,ATN2,CART,POLAR,GGG,GMS,DOT,NORM,CROSS       *)
(* MJD, CALDAT                                               *)
(* EQUECL,PMATECL, PRECART                                   *)
(* SUN200                                                    *)
(* FIND_ETA, ELEMENT                                         *)
(*-----------------------------------------------------------*)

(*-----------------------------------------------------------*)
(* START : Считывание данных и предварительная обработка дат *)
(*         наблюдений                                        *)
(* Выдача:                                                   *)
(*   RSUN: матрица трех векторов положения Солнца в          *)
```

```
(*         эклиптических координатах                              *)
(* E   : матрица трех единичных векторов наблюдаемых положений    *)
(* JD  : юлианские даты трех наблюдений                           *)
(* TEQX: равноденствие векторов в RSUN и E                        *)
(*         (в юлианских столетиях с J2000)                        *)
(*----------------------------------------------------------------*)

PROCEDURE START (VAR HEADER: CHAR80;
                 VAR RSUN,E: MAT3X; VAR JD0: REAL3; VAR TEQX: REAL);

  VAR DAY,MONTH,YEAR,G,M,I   : INTEGER;
      UT,S,DUMMY             : REAL;
      EQX0,EQX,TEQX0         : REAL;
      LS,BS,RS,LP,BP,RA,DEC,T: REAL3;
      A,AS                   : REAL33;
      ORBINP                 : TEXT;

  BEGIN

    (* открытие файла вводимых данных                             *)

    RESET(ORBINP);                                (* Standard Pascal  *)
(* ASSIGN(ORBINP,'ORBINP.DAT'); RESET(ORBINP);*)  (* Turbo Pascal     *)
  (* RESET(ORBINP,'ORBINP.DAT'); *)               (* ST Pascal plus   *)

    (* чтение данных из файла  ORBINP                             *)

    FOR I:=1 TO 80 DO                             (* заголовки        *)
      IF NOT(EOLN(ORBINP)) THEN READ(ORBINP,HEADER[I])
         ELSE HEADER[I]:=' ';
    READLN(ORBINP);
    FOR I := 1 TO 3 DO                            (* 3 наблюдения     *)
      BEGIN
        READ   (ORBINP,DAY,MONTH,YEAR,UT);        (* дата             *)
        READ   (ORBINP,G,M,S); GGG(G,M,S,RA[I]);  (* RA               *)
        READLN(ORBINP,G,M,S); GGG(G,M,S,DEC[I]);  (* Dec              *)
        RA[I]  := RA[I]*15.0;
        JD0[I] := 2400000.5+MJD(DAY,MONTH,YEAR,UT);
        T[I]   := (JD0[I]-2451545.0)/36525.0;
      END;
    WRITELN;
  READLN(ORBINP,EQX0); TEQX0:=(EQX0-2000.0)/100.0; (* равнод.        *)

    (* желаемое равноденствие элементов орбиты                    *)

    READ(ORBINP,EQX ); TEQX :=(EQX -2000.0)/100.0;
```

```
    (* вычисление исходных данных для определения орбиты          *)

    PMATECL(TEQX0,TEQX,A);
    FOR I:=1 TO 3 DO
     BEGIN
      CART    (1.0,DEC[I],RA[I],E[I,X],E[I,Y],E[I,Z]);
      EQUECL(         TEQX0,E[I,X],E[I,Y],E[I,Z]);
      PRECART( A     ,E[I,X],E[I,Y],E[I,Z]);
      POLAR  (E[I,X],E[I,Y],E[I,Z],DUMMY,BP[I],LP[I]);
      PMATECL(  T[I],TEQX,AS);
      SUN200           (T[I],LS[I],BS[I],RS[I]);
      CART    (RS[I],BS[I],LS[I],RSUN[I,X],RSUN[I,Y],RSUN[I,Z]);
      PRECART(AS,RSUN[I,X],RSUN[I,Y],RSUN[I,Z]);
    END;

WRITELN(' ORBDET:Вычисление орбиты по трем наблюдениям          ');
WRITELN('                          Версия 12.02.89               ');
WRITELN('          (c) 1989 Thomas Pfleger, Oliver Montenbruck   ');
   WRITELN; WRITELN;
   WRITELN(' Протокол определения орбиты   ');
   WRITELN;
   WRITE ('    '); FOR I:=1 TO 78 DO WRITE(HEADER[I]); WRITELN;
   WRITELN;
   WRITELN(' исходные данные ',
          '(эклиптические   геоцентрические   координаты   (град.)');
   WRITELN;
WRITELN('  юлианская дата ',JD0[1]:12:2,JD0[2]:12:2,JD0[3]:12:2);
WRITELN(' долгота Солнца   ',    LS[1]:12:2,    LS[2]:12:2,    LS[3]:12:2);
WRITELN(' долгота планеты/кометы', LP[1]:12:2, LP[2]:12:2,
                                             LP[3]:12:2);
WRITELN(' широта планеты/кометы', BP[1]:12:2, BP[2]:12:2,
                                             BP[3]:12:2);
   WRITELN; WRITELN;

  END;

(*-----------------------------------------------------------*)
(* DUMPELEM: Выдача элементов орбиты (экран)                  *)
(*-----------------------------------------------------------*)
PROCEDURE DUMPELEM(TP,Q,ECC,INC,LAN,AOP,TEQX:REAL);

  VAR DAY,MONTH,YEAR:INTEGER;
      MODJD,UT      : REAL;

  BEGIN
    MODJD:=TP*36525.0+51544.5;
    CALDAT( MODJD, DAY,MONTH,YEAR,UT);
```

```
      WRITELN(' элементы орбиты    ',
             ('  равноденствие         ','J',100.0*TEQX+2000.0:8:2,')');
    WRITELN;
    WRITELN('  момент прохождения перигелия tp ',
             DAY:2,'.',MONTH:2,'.',YEAR:4,UT:8:4,'h',
            '  (JD',MODJD+24 0 0 0 00.5:11:2,')');
    WRITELN(' расст.до перигелия   q[a.e.] ',    Q:12:6);
    WRITELN(' большая полуось      a[a.e.] ',    Q/(1-ECC):12:6);
    WRITELN(' эксцентриситет       e       ',   ECC:12:6);
    WRITELN(' наклон орбиты        i       ',   INC:10:4,        град.');
    WRITELN(' восход. узел         Omega ',     LAN:10:4,'       град.');
    WRITELN(' долгота перигелия    pi    ',   AOP+LAN:10:4,'     град.');
    WRITELN(' аргумент перигелия   omega ',     AOP:10:4,'       град.');
    WRITELN;
  END;

(*------------------------------------------------------------------*)
(* SAVEELEM: вывод элементов орбиты (в файл)                         *)
(*------------------------------------------------------------------*)
PROCEDURE SAVEELEM (TP,Q,ECC,INC,LAN,AOP,TEQX:REAL;
                                                 HEADER: CHAR80);

  VAR I,DAY,MONTH,YEAR:INTEGER;
      MODJD,UT          : REAL;

  BEGIN
    REWRITE(ORBOUT);                          (* Standard Pascal     *)
(* ASSIGN(ORBOUT,'ORBOUT.DAT'); REWRITE(ORBOUT);                      *)
                                              (* Turbo Pascal        *)
 (* REWRITE(ORBOUT,'ORBOUT.DAT'); *)          (* ST Pascal plus      *)

    MODJD:=TP*36525.0+51544.5;
    CALDAT( MODJD, DAY,MONTH,YEAR,UT);
    WRITE (ORBOUT,YEAR:5,MONTH:3,(DAY+UT/24.0):7:3,'!':6);
 WRITELN(ORBOUT,' мом.прохожд.периг. T0 (y m d.d) = JD ',
                  (MODJD+2400000.5):12:3);
   WRITELN(ORBOUT, Q :12:6,'!': 9,' q   ( a  =',Q/(1-ECC):10:6,'   )');
    WRITELN(ORBOUT,ECC:12:6,'!':  9,'  e  ');
    WRITELN(ORBOUT,INC:10:4,'!':11,'   i   ');
    WRITELN(ORBOUT,LAN:10:4,'!':11,'  долгота  восх.  узла  ');
    WRITELN(ORBOUT,AOP:10:4,'!':11,
          ' арг. перигелия ( долг. периг. = ',AOP+LAN:9:4,' )');
    WRITELN(ORBOUT,TEQX*100.0+2000.0:8:2,'!':13,'    равнод.   (J)');
    WRITE  (ORBOUT,'!  ');
    FOR I:=1 TO 78 DO WRITE(ORBOUT,HEADER[I]);
    RESET(ORBOUT); (* файл закрыт      *)
  END;
(*------------------------------------------------------------------*)
```

```
(*  RETARD: Поправка за время распространения света                 *)
(*     JD0: моменты наблюдения (t1',t2',t3') (юлианские даты)       *)
(*     RHO: три геоцентрических расстояния (а.е.)                   *)
(*      JD: моменты испускания света (t1,t2,t3) (юлианские даты)    *)
(*     TAU: три промежутка времени                                  *)
(*------------------------------------------------------------------*)
PROCEDURE RETARD ( JD0,RHO: REAL3; VAR JD,TAU: REAL3);
  CONST KGAUSS = 0.01720209895;   A = 0.00578;
  VAR  I : INTEGER;
  BEGIN
   FOR I :=1 TO 3 DO JD[I]:=JD0[I]-A*RHO[I];
   TAU[1] :=KGAUSS*(JD[3]-JD[2]);       TAU[2]:=KGAUSS*(JD[3]-JD[1]);
   TAU[3] :=KGAUSS*(JD[2]-JD[1]);
  END;
(*------------------------------------------------------------------*)
(* GAUSS: итерации по упрощенному методу Гаусса                     *)
(*                                                                  *)
(* RSUN: три вектора геоцентрического положения Солнца              *)
(* E    : три единичных вектора направления геоцентрических         *)
(*        наблюдений                                                *)
(* JD0  : три момента наблюдения (юлианские даты)                   *)
(* TP   : момент прохождения перигелия (в юл. столетиях с J2000)    *)
(* Q    : расстояние до перигелия                                   *)
(* ECC  : эксцентриситет                                            *)
(* INC  : наклон орбиты                                             *)
(* LAN  : долгота восходящего узла                                  *)
(* AOP  : аргумент перигелия                                        *)
(*------------------------------------------------------------------*)

PROCEDURE GAUSS( RSUN,E: MAT3X; JD0:REAL3;
                 VAR TP,Q,ECC,INC,LAN,AOP: REAL );

  CONST EPS_RHO =1.0E-8;

  VAR I,J               : INTEGER;
      S                 : INDEX;
      RHOOLD,DET        : REAL;
      JD,RHO,N,TAU,ETA: REAL3;
      DI                : VECTOR;
      RPL               : MAT3X;
      DD                : REAL33;

  BEGIN

    (* вычисление исходного приближения для n1 и n3      *)

    N[1]:=(JD0[3]-JD0[2])  /  (JD[3]-JD[1]);    N[2]:=-1.0;
    N[3]:=(JD0[2]-JD0[1])  /  (JD[3]-JD[1]);
```

```
(* вычисление матрицы D и ее детерминанта (det(D) = e3.d3)          *)
  CROSS(E[2],E[3],DI); FOR J:=1 TO 3 DO DD[1,J]:=DOT(DI,RSUN[J]);
  CROSS(E[3],E[1],DI); FOR J:=1 TO 3 DO DD[2,J]:=DOT(DI,RSUN[J]);
  CROSS(E[1],E[2],DI); FOR J:=1 TO 3 DO DD[3,J]:=DOT(DI,RSUN[J]);
    DET :=    DOT(E[3],DI);

WRITELN;
     WRITELN(' итерация геоцентрического расстояния rho [a.e.] ');
WRITELN;

    RHO[2]:=0,

(* итерации до тех пор, пока rho[2] не перестанет меняться          *)

    RHO[2]:=0;

    REPEAT

      RHOOLD:=RHO[2];

 (* геоцентрическое расстояние rho по n1 и n3        *)
      FOR I := 1 TO 3 DO
  RHO[I]:=( N[1]*DD[I,1] - DD[I,2] + N[3]*DD[I,3] ) / (N[I]*DET);

  (* поправка за запаздывание и вычисление разности моментов        *)
     RETARD (JD0,RHO,JD,TAU);

     (* гелиоцентрические векторы        *)
     FOR I      := 1 TO 3 DO
      FOR S     := X TO Z DO
        RPL[I,S]:=      RHO[I]*E[I,S]-RSUN[I,S];

     (* отношение сектор-треугольник eta[i] *)
     ETA[1]:=FIND_ETA( RPL[2], RPL[3], TAU[3] );
     ETA[2]:=FIND_ETA( RPL[1], RPL[3], TAU[2] );
     ETA[3]:=FIND_ETA( RPL[1], RPL[2], TAU[1] );

     (* уточнение отношения сектор-треугольник      *)
     N[1] := ( TAU[1]/ETA[1] ) / (TAU[2]/ETA[2]);
     N[3] := ( TAU[3]/ETA[3] ) / (TAU[2]/ETA[2]);
     WRITELN('     rho','     ':16,RHO[1]:12:8,RHO[2]:12:8,RHO[3]:12:8);

    UNTIL ( ABS(RHO[2]-RHOOLD) < EPS_RHO );

    WRITELN; WRITELN(' гелиоцентрические расстояния [a.e.]:');
     WRITELN;
    WRITELN( '   r   ',' ':16,
```

```
        NORM(RPL[1]):12:8,NORM(RPL[2]):12:8,NORM(RPL[3]):12:8);
    WRITELN; WRITELN;

(* элементы орбиты, определенные по первому и                    *)
                                        (* третьему наблюдениям   *)

    ELEMENT ( JD[1],JD[3],RPL[1],RPL[3], TP,Q,ECC,INC,LAN,AOP );

  END;
(*-----------------------------------------------------------    *)

BEGIN

  START(HEADER,RSUN,E,JD0,TEQX);

  GAUSS(RSUN,E,JD0,TP,Q,ECC,INC,LAN,AOP);

  DUMPELEM(TP,Q,ECC,INC,LAN,AOP,TEQX);
  SAVEELEM(TP,Q,ECC,INC,LAN,AOP,TEQX,HEADER);

  (* проверка решений          *)

  WRITELN;
  IF  (DOT(E[2],RSUN[2])>0)  THEN
    WRITELN ('внимание: наблюдения вблизи соединения;',
             'возможны двойные решения');
  IF (ECC>1.1) THEN
    WRITELN ('внимание: вероятно, решение нефизично (e>1.1) ');
  IF ( (ABS(Q-0.985)<0.1) AND (ABS(ECC-0.015)<0.05) ) THEN
    WRITELN (' внимание: вероятно, решение—орбита Земли');

END.
(*-----------------------------------------------------------    *)
```

После вызова ORBDET от пользователя не требуется никакого ввода, так как вся информация содержится в файле ORBDET(.DAT). Для приведенных выше значений получают следующий протокол:

```
          ORBDET: Определение орбиты по трем наблюдениям
                          Версия 12.02.89
              (c) 1989 Thomas Pfleger, Oliver Montenbruck
 Протокол определения орбиты
Церера (пример Гаусса)
исходные данные (эклиптические геоцентрические координаты (град.))
юлианская дата     2380570.51  2380704.42   2380830.35
долгота Солнца        162.91      297.21        61.95
```

долгота планеты/кометы	95 54	99.82	118.10
широта планеты/кометы	-0.99	7.28	7.65

итерация геоцентрического расстояния rho [a.e.]

rho	3 17388654	1.55960584	3.26891774
rho	2.97212839	1.61606981	3.04726038
rho	2.92283402	1.63070368	2.98636286
rho	2.90839087	1.63497331	2.96755085
rho	2.90390666	1.63628877	2.96157698
rho	2.90248578	1.63670369	2.95966629
rho	2.90203220	1.63683587	2.95905411
rho	2.90188709	1.63687808	2.95885777
rho	2.90184056	1.63689162	2.95879486
rho	2.90182566	1 63689595	2.95877468
rho	2.90182089	1.63689734	2.95876820
rho	2.90181936	1 63689778	2.95876612
rho	2.90181886	1.63689792	2.95876545
rho	2 90181871	1.63689797	2.95876524
rho	2 90181866	1.63689798	2.95876517
rho	2.90181864	1.63689799	2.95876515
гелиоцентрические расстояния [a.e]:			
r	2.68083158	2.58787040	2.54398482

элементы орбиты (равноденствие J 1806.00)

момент прохождения перигелия tp 26.6.1806 0.9118h (JD 2380865.54)

расст. до перигелия	q[a.e]	2.541677	
большая полуось	a[a.e.]	2 767167	
эксцентриситет	e	0.081488	
наклон орбиты	i	10.6178	град.
восход. узел	Omega	80.9788	град.
долгота перигелия	pi	147 0171	град.
аргумент перигелия	omega	66.0383	град.

Кроме того, элементы орбиты запоминаются в массиве ORBOUT.DAT:

```
1806  6 28.038 !  мом.прохожд.периг. T0 (y  m  d.d)= JD 2380865.538
    2.541677  !  q (a = 2.767167)
    0.081488  !  e
   10.6178    !  i
   80.9788    !  долгота восх. узла
   66 0383    !  арг. перигелия ( долг. периг. = 147.0171)
1806.00       !  равнод. (J)

! Церера (пример Гаусса)
```

Их можно ввести в программу COMET (гл.4), чтобы убедиться, что наблюдаемые положения воспроизведены правильно:

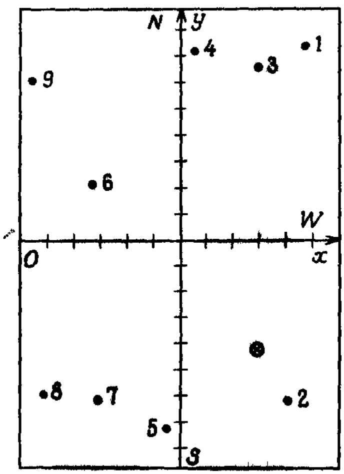
Дата	h	L	l	b	r	RA			Dec			Расст.
						h	m	s	°	′	″	(а.е.)
6.9.1805	0.2	162.9	75.2	-1.1	2 681	6	23	57 5	22	21	27	2 901867
17.1.1806	22.1	297.2	106.4	4.6	2.588	6	45	14.7	30	21	24	1.636882
23.5.1806	20.4	61.9	137.7	8.9	2.544	8	7	44.6	28	2	47	2.958699

Как показывают эклиптические координаты Солнца и Цереры, астероид во время наблюдения находился почти в противостоянии. Поэтому в рассмотренном примере нет опасности возникновения кратных решений.

10. АСТРОМЕТРИЯ

Фотография позволяет относительно просто определять координаты комет и планет путем сравнения с известными положениями звезд. При этом наряду со снимком требуется лишь звездный каталог и, возможно, атлас, чтобы облегчить отождествление звезд окрестностей.

Для наглядности на рис.10.1 схематически представлен снимок кометы Бредфилда 1982 г. Сравнение с звездным атласом показывает, что снимок охватывает область $\pm 4^m$ по прямому восхождению и $\pm 1^\circ$ по склонению, а его центр находится в точке ($\alpha_{1950} = 12^h\ 26^m\ 15^s$, $\delta_{1950} = 44{,}25^\circ$). Рядом с кометой удается отождествить ряд звезд, координаты которых можно найти в каталоге SAO (Smithsonian Astrophysical Observatory). Этот каталог перекрывает северное и южное небо и содержит примерно 250 000 звезд. Среднее расстояние между двумя соседними звездами каталога составляет примерно полградуса. Для определения положения небесного тела по одной фотографии нужны координаты по меньшей мере трех опорных звезд. Для получения более точных координат обычно стараются привлечь к определению координат возможно большее число окрестных звезд. Однако сначала мы рассмотрим, как воспроизводится на снимке область неба со звездами различных прямых восхождений и склонений.



Опорные звезды: 1. SAO 044166, 2. SAO 044175, 3. SAO 044177, 4. SAO 044187, 5. SAO 044199, 6. SAO 044207, 7. SAO 044208, 8. SAO 044220, 9. SAO 044221
⊗ Комета

Центр пластинки: $\alpha \approx 12^h 26^m 15^s$ $\delta \approx 44^\circ 15'$

Рис.10.1. Образец фотографии звездного поля.

10.1. Фотографическое изображение

Идеальная камера или объектив телескопа собирают лучи света звезды в одной точке поверхности фотопластинки. Она лежит за объективом на расстоянии *F*, называемом фокусным расстоянием. Точка *P*, в которой получается изображение звезды, находится в месте пересечения луча света, прошедшего через центр объектива *O*, с поверхностью пленки (рис.10.2).

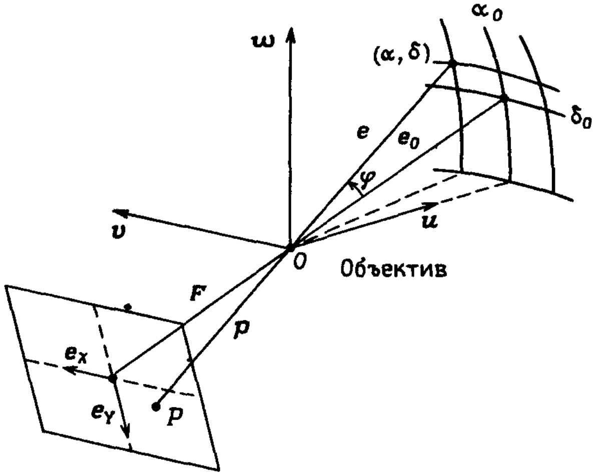


Рис.10.2. Построение изображения при фотографировании звездного поля.

В системе координат u, v, w вектор

$$\mathbf{e} = \begin{bmatrix} \cos(\delta)\cos(\alpha - \alpha_0) \\ \cos(\delta)\sin(\alpha - \alpha_0) \\ \sin(\delta) \end{bmatrix}$$

задает направление на звезду с прямым восхождением α и склонением δ. Соответственно вектор

$$\mathbf{e}_0 = \begin{bmatrix} \cos(\delta_0) \\ 0 \\ \sin(\delta_0) \end{bmatrix}$$

определяет положение (α_0, δ_0) точки на небе, куда направлена оптическая ось камеры. Векторы $\mathbf{F} = -F \cdot \mathbf{e}_0$ и $\mathbf{p} = -p \cdot \mathbf{e}$ описывают путь света от объектива к середине пластинки и к изображению звезды *P*. Они образуют между собой угол φ, для которого

$$\cos(\varphi) = \mathbf{e}_0 \cdot \mathbf{e} =$$

$$= \cos(\delta_0)\cos(\delta)\cos(\alpha - \alpha_0) + \sin(\delta_0)\sin(\delta).$$

На поверхности фотопластинки векторами

$$\mathbf{e}_X = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad \text{и} \quad \mathbf{e}_Y = \begin{bmatrix} +\sin(\delta_0) \\ 0 \\ -\cos(\delta_0) \end{bmatrix}$$

определяется координатная система для измерения снимка, оси которой ориентированы с севера на юг и с востока на запад. При помощи координат X и Y точки P, выраженных в единицах фокусного расстояния F, $\mathbf{p}$ можно представить в следующем виде:

$$\mathbf{p} = \mathbf{F} + (F \cdot X) \cdot \mathbf{e}_X + (F \cdot Y) \cdot \mathbf{e}_Y .$$

Если разложить это уравнение на компоненты, то зависимость между (α, δ) и (X, Y) может быть представлена тремя уравнениями:

$$\begin{aligned} p\cos(\delta)\cos(\alpha - \alpha_0) &= F\cos(\delta_0) - FY\sin(\delta_0), \\ p\cos(\delta)\sin(\alpha - \alpha_0) &= -FX, \\ p\sin(\delta) &= F\sin(\delta_0) + FY\cos(\delta_0), \end{aligned} \tag{10.}$$

где

$$p = |\mathbf{p}| = F\sqrt{1 + X^2 + Y^2},$$

или

$$p = F/\cos(\varphi) =$$

$$= F/(\cos(\delta_0)\cos(\delta)\cos(\alpha - \alpha_0) + \sin(\delta_0)\sin(\delta)).$$

Решение этих уравнений в сферических координатах дает

$$\alpha = \alpha_0 + \operatorname{arctg}\left\{\frac{-X}{\cos(\delta_0) - Y\sin(\delta_0)}\right\}, \tag{10.}$$

$$\delta = \arcsin\left\{\frac{\cos(\delta_0) + Y\sin(\delta_0)}{\sqrt{1 + X^2 + Y^2}}\right\}.$$

С учетом соотношений (10.2) из (10 1) получим

$$X = -\frac{\cos(\delta)\,\sin(\alpha - \alpha_0)}{\cos(\delta_0)\,\cos(\delta)\,\cos(\alpha - \alpha_0) + \sin(\delta_0)\,\sin(\delta)},$$

$$Y = -\frac{\sin(\delta_0)\,\cos(\delta)\,\cos(\alpha - \alpha_0) - \cos(\delta_0)\,\sin(\delta)}{\cos(\delta_0)\,\cos(\delta)\,\cos(\alpha - \alpha_0) + \sin(\delta_0)\,\sin(\delta)}. \qquad (10.3)$$

Оба этих преобразования могут быть легко представлены в виде подпрограмм

```
(*---------------------------------------------------------------*)
(* STDEQU  : Преобразование идеальных координат в экваториальные *)
(* RA0,DEC0: прямое восх  и склонение оптической оси   (град )   *)
(*    XX,YY  идеальные координаты                                *)
(* RA,DEC  · прямое восхождение и склонение (град.)              *)
(*---------------------------------------------------------------*)
PROCEDURE STDEQU ( RA0,DEC0,XX,YY: REAL, VAR RA,DEC: REAL);
  BEGIN
  RA = RA0 + ATN ( -XX / (CS(DEC0)-YY*SN(DEC0)) ),
  DEC = ASN ( (SN(DEC0)+YY*CS(DEC0))/SQRT(1.0+XX*XX+YY*YY) );
END,
(*---------------------------------------------------------------*)
(* EQUSTD    Преобразование экваториальных координат в идеальные *)
(* RA0,DEC0  прямое восх  и склонение оптической оси (град )     *)
(* RA,DEC  . прямое восхождение и склонение (град )              *)
(* XX,YY     идеальные координаты                                *)
(*---------------------------------------------------------------*)
PROCEDURE EQUSTD ( RA0,DEC0,RA,DEC: REAL; VAR XX,YY: REAL);
 VAR C· REAL,
 BEGIN
  C  =CS(DEC0)*CS(DEC)*CS(RA-RA0)+SN(DEC0)*SN(DEC);
  XX =- ( CS(DEC)*SN(RA-RA0) ) / C,
  YY:=- ( SN(DEC0)*CS(DEC)*CS(RA-RA0)-CS(DEC0)*SN(DEC) ) / C;
END;
(*---------------------------------------------------------------*)
```

10.2. Постоянные пластинки

Безразмерные координаты X и Y называются идеальными координатами, так как по определению они не зависят от фокусного расстояния используемой оптики и относятся к системе, проходящей через центр снимка (α_0, δ_0) и ориентированной строго параллельно меридиану. Если и при измерении фотоснимка взять за основу эту систему, то идеальные координаты получаются делением измеренных координат x и y на фокусное расстояние:

$$X = x/F \,, \qquad Y = y/F \,.$$

Вообще говоря, начало используемой координатной системы не всегда точно совпадает с точкой пересечения оптической оси с поверхностью пластинки. Малое смещение на величину (Δx, Δy) может быть учтено посредством небольшого преобразования формул:

$$X = x/F - (\Delta x)/F \,, \qquad Y = y/F - (\Delta y)/F \,.$$

Кроме того, если оси координат повернуты на угол γ относительно направления север—юг, то формулы могут быть записаны в виде

$$X = (x \cos(\gamma) - y \sin(\gamma))/F - (\Delta x)/F \,,$$

$$Y = (x \sin(\gamma) + y \cos(\gamma))/F - (\Delta y)/F \,.$$

Однако помимо сдвига и поворота системы координат имеются и другие факторы, определяющие связь между измеренными и идеальными координатами Ошибки оптики, а также возможный перекос или деформация фотопластинки, делают необходимым внесение дополнительных поправок. Поэтому для взаимного пересчета координат (x, y) и (X, Y) используют формулы в общем виде:

$$\begin{aligned} X &= a \cdot x + b \cdot y + c \,, \\ Y &= d \cdot x + e \cdot y + f \,, \end{aligned} \tag{10.4}$$

с шестью *постоянными фотопластинками* a, b, c, d, e и f. При этом постоянные пластинки заранее не известны, а определяются при помощи опорных звезд. Например, если известны экваториальные координаты $(\alpha_i, \delta_i)_{i=1,2,3}$ трех опорных звезд, то из соотношений (10.3) можно вычислить соответствующие идеальные координаты $(X_i, Y_i)_{i=1,2,3}$. учетом измеренных координат $(x_i, y_i)_{i=1,2,3}$ получаем три уравнения

$$\begin{aligned} X_1 &= x_1 a + y_1 b + c \,, \\ X_2 &= x_2 a + y_2 b + c \,, \\ X_3 &= x_3 a + y_3 b + c \,, \end{aligned} \tag{10.5}$$

которые при помощи различных преобразований можно решить относительно a, b и c. Остальные постоянные пластинки d, e, f определяют ся из соответствующих уравнений для Y_i. Фокусное расстояние фотокамеры при этом не используется. Это особенно выгодно, если используют увеличенный фотоснимок, а масштаб увеличения не известен.

10.3. Решение уравнений

Итак, при наличии трех опорных звезд с известными координатами постоянные пластинки могут быть получены решением двух систем уравнений, каждая из которых содержит три неизвестных. Так как измерения координат звезд не могут быть свободны от ошибок, при определении постоянных пластинки желательно использовать как можно больше опорных звезд. Правда, система уравнений вида

$$\begin{aligned} X_1 &= x_1\, a + y_1\, b + c\,, \\ &\ldots \\ X_n &= x_n\, a + y_n\, b + c\,, \end{aligned} \tag{10.6}$$

при $n > 3$ уравнений для трех искомых постоянных пластинки переопределена и поэтому не решается. Чтобы обойти это затруднение, в дальнейшем при решении уравнений используется *метод наименьших квадратов.*

Если нужно определить m неизвестных $s_1,\ldots, s_m$ из n уравнений

$$t_i = A_{i1} \cdot s_1 + \ldots + A_{im} \cdot s_m \qquad (i = 1,\ldots,n) \tag{10.7}$$

с заданными коэффициентами t_i и A_{ij}, то для $n > m$ нельзя ожидать, что все уравнения будут одновременно удовлетворены. Поэтому вводят так называемые остаточные уклонения

$$r_i = A_{i1} \cdot s_1 + \ldots + A_{im} \cdot s_m - t_i \qquad (i = 1,\ldots,n)\,. \tag{10.8}$$

Вместо поиска величин $s_1,\ldots,s_m$, для которых все r_i равны нулю (это означало бы, что уравнения удовлетворяются), будем минимизировать сумму квадратов остаточных уклонений

$$S = \sum_{i=1}^{n} r_i^2\,.$$

Для решения этой задачи уравнения (10.8) вначале преобразуют к эквивалентной системе

$$
\begin{aligned}
r'_1 &= A'_{11} \cdot s_1 + A'_{12} \cdot s_2 + \dots + A'_{1m} \cdot s_m - t'_1 \,, \\
r'_2 &= \qquad\qquad\quad A'_{22} \cdot s_2 + \dots + A'_{2m} \cdot s_m - t'_2 \,, \\
\vdots &= \qquad\qquad\qquad\qquad \ddots \qquad \vdots \qquad\qquad \vdots \\
r'_m &= \qquad\qquad\qquad\qquad\qquad\qquad A'_{mm} \cdot s_m - t'_m \,, \\
r'_{m+1} &= \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad - t'_{m+1} \,, \\
\vdots &= \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
r'_n &= \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad - t'_n \,,
\end{aligned}
\tag{10.9}
$$

в которой все A'_{ij} с $(i > j)$ равны нулю, что существенно упрощает использование.

Если p и q—два действительных числа и $p^2 + q^2 = 1$, то сумма квадратов остаточных уклонений не изменится, например, если r_1 заменить на $r'_1 = p \cdot r_1 - q \cdot r_2$ и r_2 заменить на $r'_2 = q \cdot r_1 + p \cdot r_2$, так как в этом случае

$$
\begin{aligned}
r_1'^2 + r_2'^2 &= p^2 r_1^2 - 2pqr_1r_2 + q^2 r_2^2 + q^2 r_1^2 + 2pqr_1r_2 + p^2 r_2^2 = \\
&= (p^2 + q^2) \cdot (r_1^2 + r_2^2)
\end{aligned}
$$

равно $r_1^2 + r_2^2$. Если p и q подобрать так, что

$$p = +A_{11}/h \quad \text{и}$$

$$q = -A_{21}/h, \quad \text{где} \quad h = \pm\sqrt{A_{11}^2 + A_{21}^2}\,,$$

то тогда A_{21} можно заменить на

$$A'_{21} = qA_{11} + pA_{21} = (-A_{21}A_{11} + A_{11}A_{21})/h = 0\,,$$

к чему мы и стремились. Продолжив такие преобразования, называемые вращением Гивенса, можно последовательно исключить $A_{31}\ A_{n1}$, $A_{32} \dots A_{n2}, \dots$, и $A_{m+1,m}\ A_{nm}$, пока, наконец, не получим систему уравнений в форме (10.9). Сумма квадратов остаточных уклонений может быть разделена на две части:

$$S = \sum_{i=1}^{m} r_i'^2 + \sum_{i=m+1}^{n} t_i'^2\,,$$

из которых только первая зависит от неизвестных s_k. Очевидно, сумма будет наименьшей, когда все r_i для $i = 1 \dots m$ обратятся в нуль. Соответствующее значение s_k получим из

$$s_m = t'_m \;/\; A'_{mm} \;,$$

$$s_k = (t'_k - \sum_{l=k+1}^{m} A'_{kl}s_l) \;/\; A'_{kk} \quad (k = m-1,\dots,1)\;. \tag{10.10}$$

Преобразование системы уравнений и ее дальнейшее решение осуществляет подпрограмма LSQFIT, с использованием которой подобные линейные задачи легко решаются.

```
(*-----------------------------------------------------------------  *)
(* LSQFIT : Решение системы уравнений                                 *)
(*          A[i,1]*s[1]+...A[i,m]*s[m] - A[i,m+1] = 0      (i=1,..,n)  *)
(*          методом наименьших квадратов с применением                *)
(*          вращения Гивенса                                          *)
(*      A:  матрица коэффициентов                                     *)
(*      N : число уравнений (ряды A)                                  *)
(*      M:  число неизвестных (столбцы A, элементы S)                 *)
(*      S : вектор решения                                            *)
(*-----------------------------------------------------------------  *)
PROCEDURE LSQFIT ( A: LSQMAT; N, M: INTEGER; VAR S: LSQVEC );

  CONST EPS = 1.0E-10;  (* точность вычислений                        *)

  VAR I,J,K:  INTEGER;
      P,Q,H:  REAL;

  BEGIN
    FOR J:=1 TO M DO (* Цикл по столбцам 1...M для A                  *)
       (* исключение элементов A[i,j] с i>j из столбца j              *)

       FOR I:=J+1 TO N DO
         IF  A[I,J]<>0.0  THEN
          BEGIN
    (* вычисление p, q и новых A[j,j] ; приравнивание A[i,j]=0        *)
            IF  (   ABS(A[J,J])<EPS*ABS(A[I,J])   )
               THEN
                  BEGIN
                   P:=0.0;    Q:=1.0;    A[J,J]:=-A[I,J];    A[I,J]:=0.0;
                  END
               ELSE
                  BEGIN
                 H:=SQRT(A[J,J]*A[J,J]+A[I,J]*A[I,J]);
                 IF  A[J,J]<0.0  THEN  H:=-H;
                 P:=A[J,J]/H;    Q:=-A[I,J]/H;    A[J,J]:=H;    A[I,J]:=0.0;
                 END;
            (*  обработка остальных строк             *)
            FOR K:=J+1 TO M+1 DO
```

```
            BEGIN
              H      :=  P*A[J,K]  -  Q*A[I,K];
              A[I,K]:=  Q*A[J,K]  +  P*A[I,K];
              A[J,K]:=  H,
            END;
          END;
    (* обратная подстановка                          *)

    FOR I    :=  M DOWNTO 1 DO
      BEGIN
        H    :=  A[I,M+1];
        FOR K:= I+1 TO M DO H:=H+A[I,K]*S[K];
        S[I]  :=  -H/A[I,I];
      END;

  END;    (* LSQFIT *)
(*---------------------------------------------------------      *)
```

При этом массив A в первых m столбцах содержит коэффициенты A_{ij} ($i = 1 \dots n$, $j = 1 \dots m$), а в остальных столбцах — величины t_i ($i = 1 \dots n$). После вызова подпрограммы S содержит искомые значения неизвестных $s_{1 \dots m}$. Для работы с LSQFIT должны быть описаны два типа

```
TYPE LSQVEC      = ARRAY[1..MDIM] OF REAL;
                LSQMAT= ARRAY[1..NDIM,1..M1DIM] OF REAL;
```

Размерности MDIM и NDIM должны быть по крайней мере такими же, как число неизвестных (m) и число уравнений (n) соответственно. M1DIM дает число столбцов A и поэтому не может быть меньше $m + 1$.

Нужно отметить, что в описываемом методе решения системы уравнений целиком запоминать матрицу A не обязательно. Отдельные коэффициенты A_{ij} могут исключаться не столбцами, а построчно. При соответствующем программировании удается эффективно решить сложную проблему большого числа уравнений. Однако предлагаемой версией LSQFIT проще пользоваться, и она вполне достаточна для решения поставленной задачи.

10.4. Программа FOTO

Программа FOTO позволяет определять точные положения звезд, комет и астероидов по фотографическим пластинкам. Пользователь при этом освобождается от громоздких вычислений, необходимых для определения идеальных координат и решения системы уравнений.

До вызова программы нужно провести некоторую предварительную работу по подготовке исходных величин. Вначале с помощью звездной карты отождествляют несколько ярких звезд и устанавливают приблизительные координаты сфотографированной области. Затем выбирают опорные звезды, экваториальные координаты которых можно найти в звездном каталоге, например в каталоге SAO. Конечно, нужно учесть собственное движение звезды за время, прошедшее между эпохой каталога и моментом получения фотографии. Опорные звезды должны быть равномерно распределены по пластинке и для проведения точных измерений по возможности должны иметь точечный вид. При наличии увеличенного снимка положения опорных звезд и исследуемого объекта легко определяются при помощи прозрачной миллиметровой бумаги, которую кладут на снимок и приблизительно ориентируют в направлении север—юг. Наконец, определяют прямое восхождение и склонение центра пластинки, нужные для вычисления идеальных координат. Так как ошибка координат центра пластинки мало влияет на результаты определения положений, достаточно взять эти значения из атласа.

Все данные вводятся в файл FOTINP(.DAT). В первой строке этот файл содержит прямое восхождение (h, m, s) и склонение (°,′,″) центра пластинки. Затем следуют данные об объектах снимка. Опорные звезды, используемые для определения постоянных пластинки, обозначаются значком * в первом столбце. После имени, которое может содержать до 11 символов, задаются измеренные координаты (x, y в мм) и экваториальные координаты (α в h, m, s, δ в °,′,″). Приведенные в следующем примере данные относятся к рис. 10.1.

ЦЕНТР				12	26	15. 0		44	15	00.0
*SAO	044166	+47.5	+73.3	12	22	40.293		45	04	33.40
*SAO	044175	+41.2	-62.1	12	23	32.960		43	23	42.59
*SAO	044177	+29.9	+65.2	12	23	56.037		44	59	16.14
*SAO	044187	+5.5	+71.1	12	25	39.308		45	04	14.33
*SAO	044199	-4.8	-72.7	12	26	43.594		43	17	28.86
*SAO	044207	-33.0	+21.0	12	28	27.655		44	27	53.23
*SAO	044208	-31.2	-62.0	12	28	27.755		43	26	04.19
*SAO	044220	-51.2	-59.7	12	29	52.083		43	28	33.76
*SAO	044221	-55.7	+60.6	12	29	56.524		44	57	27.65
Бредфилд		+29.2	-42.1							

Подпрограмма GETINP считывает и запоминает эти данные. Затем определяются идеальные координаты опорных звезд, необходимые для вычисления постоянных пластинки. Тогда, зная постоянные пластинки, можно вычислить идеальные координаты и экваториальные координаты всех объектов. Таким образом можно получить хорошую оценку ошибки вычислений. Кроме того, так можно выявить большие ошибки, связанные с неправильным отождествлением опорных звезд или неправильным измерением их положений.

```
(*-----------------------------------------------------------------*)
(*                            FOTO                                 *)
(*          Астрометрическая обработка фотопластинок               *)
(*                       Версия 11.01.89                           *)
(*-----------------------------------------------------------------*)
```

```
PROGRAM FOTO (INPUT,OUTPUT,FOTINP);

 CONST MAXDIM    = 30; (* максимальное число объектов на фото
       NAME_LENGTH=12,
       ARC        = 206264.8; (* секунд дуги в радиане *)

  TYPE  NAME_TYPE= ARRAY[1..NAME_LENGTH] OF CHAR;
        LSQVEC    = ARRAY[1..3] OF REAL;
        LSQMAT    = ARRAY[1..MAXDIM,1..5] OF REAL;
        REAL_ARRAY=ARRAY[1 .MAXDIM] OF REAL;
        NAME_ARRAY=ARRAY[1..MAXDIM] OF NAME_TYPE;

  VAR I,J,K, NREF,NOBJ, GRAD,MIN:  INTEGER;
       RA0,DEC0, A,B,C,D,E,F, SEC.  REAL;
       RA_OBS,DEC_OBS, D_RA,D_DEC:REAL;
       DET, FOC_LEN, SCALE        : REAL;
       RA,DEC, X,Y, XX,YY, DELTA : REAL_ARRAY;
       S                          : LSQVEC;
       AA                         : LSQMAT;
       NAME                       : NAME_ARRAY;
       FOTINP                     : TEXT;
(* Сюда вводятся следующие подпрограммы в указанном порядке:     *)
(* SN, CS, ATN, ASN, GGG, GMS, LSQFIT                             *)
(* EQUSTD, STDEQU                                                 *)
(*-------------------------------------------------------------   *)

(*--------------------------------------------------------------
(* GETINP. чтение введенных данных из файла
(*--------------------------------------------------------------
PROCEDURE GETINP ( VAR RA0,DEC0: REAL;VAR NOBJ: INTEGER;
          VAR NAME. NAME_ARRAY;VAR RA,DEC,X,Y: REAL_ARRAY );

  VAR  I,K,H,M:  INTEGER;
       S       : REAL;
       C       : CHAR;

  BEGIN

  WRITELN;
  WRITELN ('  FOTO: Астрометрическая обработка фотопластинок       ');
  WRITELN ('                        Версия 15.01.89                ');
  WRITELN ('        (c) 1988 Thomas Pfleger, Oliver Montenbruck    ');
  WRITELN;

    (* открытие файла для чтения                  *)
       RESET(FOTINP);                             (* Standard Pascal
(* ASSIGN(FOTINP,'FOTINP.DAT'); RESET(FOTINP);*) (* TURBO Pascal
```

```
(* RESET(FOTINP,'FOTINP.DAT'); *)                    (* ST Pascal plus     *)

   WRITELN (' файл данных: FOTINP'); WRITELN;

   (* считывание координат центра пластинки    *)
   FOR K:=1 TO NAME_LENGTH DO READ(FOTINP,C);
   READ  (FOTINP,H,M,S);  GGG(H,M,S,RA0);  RA0:=15.0*RA0;
   READLN(FOTINP,H,M,S); GGG(H,M,S,DEC0);

(* чтение имен, координат на пласт. (и экваториальных коорд.)          *)
   I   :=  0;
   REPEAT
     I:=   I+1;
FOR K:=1 TO NAME_LENGTH DO READ(FOTINP,NAME[I][K]); (* имя  *)
     IF    NAME[I][1]='*'
       THEN (* опорн  звезда          *)
         BEGIN
      READ  (FOTINP,X[I],Y[I]);
    READ    (FOTINP,H,M,S);    GGG(H,M,S,RA[I]);    RA[I]:=15.0*RA[I];
            READLN(FOTINP,H,M,S); GGG(H,M,S,DEC[I]);
         END
       ELSE (* определяемый объект  *)
         BEGIN
            READLN    (FOTINP,X[I],Y[I]);    RA[I]:=0.0;    DEC[I]:=0.0;
         END,
   UNTIL EOF(FOTINP);

   NOBJ:=  I;

 END;
(*-----------------------------------------------------------------    *)

BEGIN   (* FOTO                                                          *)

  (* чтение файла данных                                               *)

  GETINP ( RA0,DEC0, NOBJ, NAME, RA,DEC,X,Y );

  (* вычисление идеальных координат опорных звезд                      *)
  (* размещение элементов матрицы AA                                   *)
  (* (столбец AA[*,5] служит как промежуточный накопитель)             *)

  J     :=0;
  FOR I:=1 TO NOBJ DO
    IF  NAME[I][1]='*'  THEN
         BEGIN
          J:=   J+1;
```

```
        EQUSTD  (  RA0,DEC0,  RA[I],DEC[I],  XX[I],YY[I]  );
        AA[J,1]:=  X[I];   AA[J,2]:=  Y[I];   AA[J,3]:=1.0;
        AA[J,4]:=-XX[I];        AA[J,5]:=-YY[I];
      END;
  NREF := J;     (* число опорных звезд                                  *)

  (* вычисление постоянных пластинки a,b,c                               *)

  LSQFIT ( AA, NREF, 3, S );   A:=S[1];    B:=S[2];    C:=S[3];

  (* вычисление постоянных пластинки d,e,f                               *)

 FOR I:=1 TO NREF DO AA[I,4]:=AA[I,5]; (* копирование столбца           *)
                                       (* A[*,5]->A[*,4]                 *)
  LSQFIT  (  AA,  NREF,  3,  S  );  D:=S[1];  E:=S[2];  F:=S[3];

(* вычисление экваториальных координат и ошибок                          *)

  FOR I:=1 TO NOBJ DO
    BEGIN
       XX[I]  :=  A*X[I]+B*Y[I]+C;
       YY[I]  :=  D*X[I]+E*Y[I]+F;
       STDEQU(  RA0,DEC0,  XX[I],YY[I],  RA_OBS,DEC_OBS  );
       IF NAME[I][1]='*' THEN (* вычисление ошибок в сек. дуги          *)
        BEGIN
          D_RA    :=  (RA_OBS-RA[I])*CS(DEC[I]);
          D_DEC   :=  (DEC_OBS-DEC[I]);
          DELTA[I]:=  3600.0 * SQRT ( D_RA*D_RA + D_DEC*D_DEC );
        END;
       RA[I]:=   RA_OBS;    DEC[I]:=   DEC_OBS;
    END;

  (* фокусное расстояние                                                 *)

  DET      :=  A*E-D*B;
  FOC_LEN:=1.0/SQRT(ABS(DET));
  SCALE  :=  ARC / FOC_LEN;

  (* выдача          *)

  WRITELN (' постоянные пластинки:'       );
  WRITELN;
  WRITELN ('  a =',a:12:8,'       b =',b:12:8,'        c =',c:12:8);
  WRITELN ('  d =',d:12:8,'       e =',e:12:8,'        f =',f:12:8);
  WRITELN;
WRITELN ('эффективное фокусное расстояние и масштаб изображения:'),
  WRITELN;
```

```
  WRITELN ('  F =',  FOC_LEN:9:2,'  мм');
  WRITELN ('  m =',  SCALE:7:2,'   "/мм');
  WRITELN;
  WRITELN  ('  координаты:');
  WRITELN;
  WRITELN    ('    имя':11,    'x':9,'y':7,'X':8,'Y':8,
            'RA':12,   'DEC':13,   'ошибка':11    );
  WRITELN  ('мм':20,  'мм':7,  '   ':23,
            'h   m   s   ', '  °  '  "  ':12, '  "  ': 7  );

  FOR I:=1 TO NOBJ DO
    BEGIN
      WRITE('   '); FOR K:=1 TO NAME_LENGTH DO WRITE(NAME[I][K]);
      WRITE(X[I].7:1,Y[I]:7:1,XX[I]:9:4,YY[I]:8:4);
      GMS(RA[I]/15.0,GRAD,MIN,SEC);   WRITE(GRAD:5,MIN:3,SEC:6:2);
      GMS(DEC[I],GRAD,MIN,SEC);  WRITE(GRAD:4,MIN:3,SEC:5:1);
      IF  NAME[I][1]='*'  THEN  WRITE(DELTA[I]:7:1);
      WRITELN;
    END;
  WRITELN;

END. (* FOTO     *)
(*-----------------------------------------------------------*)
```

С данными приведенного выше примера FOTO дает следующие результаты:

FOTO: Астрометрическая обработка фотопластинок
Версия 15.01.89
(c) 1988 Thomas Pfleger, Oliver Montenbruck

файл данных: FOTINP
постоянные пластинки:
a = 0.00021648 b = 0.00000791 c = 0.00013818
d =-0.0000789 e = 0.00021643 f = -0.00103320

эффективное фокусное расстояние и масштаб изображения:

F = 4616.85 мм
m = 44.68"/мм
координаты:

имя	x	y	X	Y	RA			Dec			ошибка
	мм	мм			h	m	s	°	'	"	"
*SAO 044166	47.5	73.3	0.0110	0.0145	12	22	40.82	45	4	29.1	7.0
*SAO 044175	41.2	-62.1	0.0086	-0.0148	12	23	32.90	43	24	0.7	18.1
*SAO 044177	29.9	65.2	0.0071	0.0128	12	23	56.46	44	59	3.5	13.4
*SAO 044187	+5.5	71.1	0.0019	0.0143	12	25	38.18	45	4	11.4	12.3
*SAO 044199	-4.8	-72.7	-0.0015	-0.0167	12	26	42.88	43	17	29.4	7.8

*SAO 044207	-33.0	21.0	-0.0068	0.0038	12	28	26.78	44	27	53.3	9.4
*SAO 044208	-31.2	-62.0	-0.0071	-0.0142	12	28	29.56	43	26	5.1	19.7
*SAO 044220	-51.2	-59.7	-0.0114	-0.0136	12	29	51.32	43	28	12.5	22.8
*SAO 044221	-55.7	60.6	-0.0114	0.0125	12	29	57.32	44	57	49.1	23.1
Бредфилд	29.2	-42.1	0.0061	-0.0104	12	24	18.57	43	39	16.4	

После постоянных пластинки выдается эффективное фокусное расстояние (т.е. фокусное расстояние камеры, умноженное на увеличение) и масштаб изображения. Затем для каждого отдельного объекта следуют измеренные прямоугольные координаты (x, y), вычисленные идеальные координаты (X, Y) и экваториальные координаты — прямое восхождение и склонение. Для опорных звезд получаются характерные отличия от каталожных значений 10—20″. Это соответствует ошибке измерения примерно 1/4—1/2 мм, чего и можно было ожидать при определении положений с помощью миллиметровки. Более точные результаты могут быть получены при помощи специального измерительного стола.

В этом примере не учитывались собственные движения за время между эпохой каталога SAO (1950.0) и моментом получения снимка (1982.8). Для избранных нами звезд сравнения они составляют максимум 7″, т. е. они меньше, чем ошибки наблюдений.

ПРИЛОЖЕНИЯ

П.1. Указания к применению различных типов компьютеров

Хотя программы, написанные на языке Паскаль, могут быть использованы на различных компьютерах, между отдельными вариантами языка существуют некоторые различия. Частично это обусловлено необходимостью приспособить компилятор к архитектуре машины или операционной системы для оптимального использования возможностей процессора. В следующем разделе обсуждаются важнейшие проблемы, которые могут возникнуть при переносе программ с одного компьютера на другой. Более подробные указания по этому поводу можно найти в руководстве, прилагаемом к каждому компьютеру.

Целые числа

В зависимости от типа компьютера особенно сильно различаются области значений чисел типа REAL и INTEGER и точности REAL-арифметики. И то, и другое существенно зависит от числа байтов[1], необходимых для записи числа. Наиболее распространены 2- и 4-байтные INTEGER-числа с областью значений −32 767...+32 767 и −2 147 483 647...+2 147 483 647. Различные компиляторы предлагают оба представления, но 2-байтные числа обрабатываются значительно быстрее. Помимо типа INTEGER, может использоваться, например, тип LONG_INTEGER.

Для приведенных здесь программ, вообще говоря, неважно, какое представление целых чисел подразумевается под именем типа INTEGER. Единственное исключение составляет выделение целой части числа с плавающей запятой, для которого обычно используется стандартная функция

FUNCTION TRUNC (X: REAL). INTEGER;

Если здесь аргумент выходит за область значений INTEGER, то это приводит к сбою во время работы программы. Поэтому для подпрограмм

CALDAT и LMST

необходимо вместо TRUNC использовать функцию

FUNCTION LONG_TRUNC (X· REAL): LONG_INTEGER;

[1] 1 байт = 8 бит. Бит—наименьшая единица памяти компьютера, она содержит двоичную цифру 0 или 1.

если используемый компилятор стандартно использует 2-байтовые INTEGER-числа.

Широко распространенный язык Turbo Pascal, хотя и не имеет 4-байтовой целочисленной арифметики, предлагает функцию

FUNCTION INT (X: REAL): REAL;

которую можно использовать вместо TRUNC в названных программах.

Выход за пределы области чисел INTEGER при использовании функции TRUNC возможен как в подпрограммах

SUN200, MINI_MOON, NUTEQU и MOON,

так и в IMPROVE в программе NEWMOON при вычислениях, относящихся к датам до −400 года. И в этом случае возникающая трудность преодолима заменой TRUNC на LONG_TRUNC или INT.

Вычисления с плавающей запятой

Для большинства программ, рассмотренных в этой книге, точность вычислений с плавающей запятой не имеет большого значения. Коротко поясним это утверждение. При вычислении координат Луны и планет исходные модели и константы позволяют добиться точности 1″. Это соответствует числу с правильными 5—6 десятичными знаками с учетом 2—3 знаков, потерянных при округлении в расчетах по различным формулам. Если работать с 10—11 знаками, что возможно на всех современных вычислительных машинах, то нечего опасаться заметных ошибок. Однако мы не советуем работать с точностью меньше 8 знаков, так как в этом случае могут возникнуть трудности в подпрограммах календарных вычислений.

В некоторых подпрограммах используются константы, рассчитанные на машинную точность $\varepsilon \approx 10^{-11}$.

Подпрограмма	Константы		Значение
ASN	EPS	=1.0E-7	$0.1\sqrt{\varepsilon}$
ACS	EPS	=1.0E-7	$0.1\sqrt{\varepsilon}$
ECCANOM	EPS	=1.0E-11	ε
HYPANOM	EPS	=1.0E-10	10ε
STUMPFF	EPS	=1.0E-12	0.1ε
PARAB	EPS	=1.0E-9	$10^2\varepsilon$
FIND_ETA	DELTA	=1.0E-9	$10^2\varepsilon$
GAUSS	EPS_RH0	=1.0E-8	$10^3\varepsilon$
LSQFIT	EPS	=1.0E-10	10ε

При необходимости эти значения можно приспособить к любой имеющейся в распоряжении точности. Однако модификация нужна лишь в т случае, если (при низкой точности) итерация в подпрограммах ECCAMON, HYPANOM, PARAB, FIND_ETA или GAUSS не сходится.

При компиляторах, поддерживающих обычную и двойную точность, простой точности (7. знаков), вообще говоря, не достаточно. Чтобы обеспечить более высокую точность, обычно следует иначе описывать тип (например, DOUBLE или LONG_FLOAT вместо REAL) для чисел с плавающей запятой. Во многих случаях можно избежать существенной переработки программ, вводя глобальное определение

TYPE REAL = DOUBLE;

кроме того, нужно учесть, что некоторые буквенные константы нужно специально объявлять с двойной точностью. В большинстве случаев это означает, что нужно использовать экспоненциальную форму записи, где E заменяется на D, например:

PI = 3 141592654D0 вместо PI = 3.141592654.

Файлы

Управление внешними файлами потребовало почти во всех разработках Паскаля расширения стандартного языка, недостаточного для систем управления данными, используемых на современных компьютерах. В первую очередь это относится к соотношению между именем файла переменных внутри программы и внешним именем, под которым файл фигурирует в операционной системе. Программа, читающая два числа из файла и затем выдающая их, на стандартном Паскале выглядела бы так:

```
   PROGRAM LESEN ( INPUT, OUTPUT, F);
      VAR X,Y: REAL;
        F    : TEXT;
BEGIN
   RESET( F )
   READ ( F, X,Y);
   WRITE( X,Y );
END;
```

Файловая переменная F указана в первой строке программы, так как соответствующий файл существует всегда, а не только во время работы программы. Чтобы теперь прочитать нужные данные, например, из файла с именем DATEN.DAT, существует несколько вариантов команды RESET, самые распространенные из которых приведены в следующей таблице.

Компилятор	Команда, открывающая файл для считывания данных
Turbo Pascal	ASSIGN(F,'DATEN.DAT'); RESET(F);
ST Pascal plus	RESET(F,'DATEN.DAT');
DEC Vax	OPEN(F,'DATEN.DAT'); RESET(F);

Точно так же можно открыть файл для записи различными модификациями команды REWRITE.

Процедуры как параметры

Паскаль позволяе передавать в подпрограммы процедуры и функции в качестве параметров. Это преимущество использовано в подпрограмме T_FIT_LBR для разложения координат в полином Чебышева. Так как некоторые компиляторы (особенно ранние версии Turbo Pascal) не имеют такой возможности, подпрограммы

T_FIT_MOON и T_FIT_SUN

при необходимости нужно перепрограммировать. Сделать это довольно просто. Нужно директивную часть названных процедур заменить на спецификацию и директивную часть T_FIT_LBR, а затем заменить имена переменных L_POLY и B_POLY на RA_POLY и DE_POLY, а L и B на RA и DE. Измененные соответствующим образом версии T_FIT_MOON и T_FIT_SUN записаны на дискету, приложенную к этой книге[1].

Имена переменных

В программах этой книги часто используются имена переменных и процедур, содержащие знак подчеркивания (_), который *не* соответствует стандартам ANSI или ISO. Однако нам не хотелось бы от него отказываться, поскольку его применение способствует ясности и легкости восприятия при чтении.

Сегодня почти все компиляторы допускают такое расширение допустимых буквенных сочетаний, и у большинства пользователей никаких трудностей не возникнет. В случаях сомнения можно заменить этот символ на другой, например X.

П.2. Список подпрограмм

Представленный список объединяет те функции и процедуры, которые не привязаны к какой-то одной основной программе; их особенно удобно использовать для создания новых программ. Наряду с коротким описанием приводятся номера разделов, в которых напечатан исходный текст, и названия подпрограмм, которые используются в данной подпрограмме

ABERRAT	8.1	звездная аберрация
ACS	2.1	arccos (град.)
APPARENT	8.5	видимые координаты (CART, POLAR, PRECART)

[1] **В оригинальном издании.** —*Прим. ред.*

ASN	2.1	arcsin (град.)
ATN	2.1	arctg (град.)
ATN2	2.1	arctg (град., в правильном квадранте)
CALDAT	2.2	календарные даты
CART	2.1	прямоугольные координаты из полярных координат (CS, SN)
CROSS	9.1	векторное произведение двух векторов
CS	2.1	cos (град.)
CUBR	2.1	корень кубический
DOT	9.1	скалярное произведение двух векторов
ECCANOM	4.3	эксцентрическая аномалия (эллипс)
ECLEQU	2.3	преобразование эклиптической системы координат в экваториальную (CS, SN)
ELEMENT	9.1	определение элементов орбиты по двум векторам положений (ATN2, CROSS, DOT, NORM, POLAR)
ELLIP	4.3	положение и скорость на эллиптической орбите (ECCANOM)
EQUECL	2.3	преобразование экваториальной системы координат в эклиптическую (CS, SN)
EQUSTD	10.1	определение идеальных координат по экваториальным (CS, SN)
ETMINUT	7.3	аппроксимация разности ET—UT
FIND_ETA	9.1	отношение сектор—треугольник
GAUSVEC	4 5	векторы Гаусса (CS, SN)
GEOCEN	5 5	геоцентрические координаты (CS, SN)
GGG	2 1	перевод минут и секунд в доли градуса
GMS	2.1	перевод долей градуса в минуты и секунды
HYPANOM	4.3	эксцентрическая аномалия (гипербола)
HYPERB	4.3	положение и скорость на гиперболической орбите (HYPANOM)
JUP200	5.3	гелиоцентрические координаты Юпитера
KEPLER	4 5	положение и скорость кометы (задача двух тел при любой форме орбиты) (ELLIP, HYPERB, PARAB, ORBECL)
LMST	3.3	среднее местное звездное время
LSQFIT	10 3	метод наименьших квадратов
MAR200	П.3	гелиоцентрические координаты Марса
MER200	П.3	гелиоцентрические координаты Меркурия
MINI_MOON	3.2	координаты Луны с малой точностью
MINI_SUN	3.2	координаты Солнца с малой точностью
MJD	2.2	модифицированные юлианские даты
MOON	6.2	более точные эклиптические координаты Луны
MOONEQU	6.2	более точные экваториальные координаты Луны (CART, ECLEQU, MOON, NUTEQU, POLAR)
NEP200	П 3	гелиоцентрические координаты Нептуна
NORM	9.1	длина вектора (DOT)
NUTEQU	5.4	нутация в экваториальных координатах
ORBECL	4 5	эклиптические координаты точки в плоскости орбиты

PARAB	4.4	положение и скорость на параболической орбите (STUMPFF)
PLU200	П.3	гелиоцентрические координаты Плутона
PMATECL	2.4	матрица прецессии (эклиптические координаты) (CS, SN)
PMATEQU	2.4	матрица прецессии (экваториальные координаты) (CS, SN)
PN_MATRIX	8.1	матрица прецессии и нутации (экваториальные координаты) (PMATEQU, NUTEQU)
POLAR	2.1	преобразование прямоугольных координат в полярные (ATN2)
PRECART	2.4	прецессия в прямоугольных координатах
QUAD	3.7	квадратичная интерполяция по трем точкам
SAT200	П.3	гелиоцентрические координаты Сатурна
SITE	8.3	координаты точки на поверхности Земли (CS, SN)
SN	2.1	sin (град.)
STDEQU	10.1	преобразование идеальных координат в экваториальные (CS, SN)
STUMPFF	4.4	функции Штумпфа
SUN200	2 5	эклиптические координаты Солнца
SUNEQU	7.5	экваториальные координаты Солнца (CART, ECLEQU, NUTEQU, SUN200)
T_EVAL	6.3	вычисление полинома Чебышева
T_FIT_LBR	6.3	аппроксимация полиномами Чебышева
T_FIT_MOON	6 3	аппроксимация координат Луны полиномами Чебышева (MOONEQU, T_FIT_LBR)
T_FIT_SUN	7.5	аппроксимация координат Солнца полиномами Чебышева (SUNEQU, T_FIT_LBR)
TN	2.1	tg (град.)
URA200	П.3	гелиоцентрические координаты Урана
VEN200	П.3	гелиоцентрические координаты Венеры

П.3. Подпрограммы вычисления гелиоцентрических положений планет

Меркурий

```
(*-----------------------------------------------------------------*)
(* MER200: Меркурий; эклиптические координаты L,B,R(град., а.е.)   *)
(*         равноденствие даты                                       *)
(*         (T· время в юлианских столетиях с J2000)                 *)
(*         (   = (JED-2451545.0)/36525    )                         *)
(*-----------------------------------------------------------------*)
PROCEDURE MER200(T:REAL;VAR L,B,R:REAL);
  CONST P2=6.283185307;
  VAR C1,S1:  ARRAY [-1..9] OF REAL;
        C,S:  ARRAY [-5..0] OF REAL;
```

```
         M1,M2,M3,M5,M6:REAL;
         U,V,  DL,DR,DB :  REAL;
         I              :  INTEGER;

   FUNCTION FRAC(X:REAL):REAL;
    BEGIN  X:=X-TRUNC(X);  IF (X<0) THEN X:=X+1.0;  FRAC:=X   END;

   PROCEDURE ADDTHE(C1,S1,C2,S2:REAL; VAR C,S:REAL);
    BEGIN  C:=C1*C2-S1*S2;      S:=S1*C2+C1*S2;
    END;

   PROCEDURE TERM(I1,I,IT:INTEGER;DLC,DLS,DRC,DRS,DBC,DBS:REAL);
    BEGIN
      IF   IT=0   THEN   ADDTHE(C1[I1],S1[I1],C[I],S[I],U,V)
               ELSE BEGIN  U:=U*T;  V:=V*T END;
      DL:=DL+DLC*U+DLS*V; DR:=DR+DRC*U+DRS*V; DB:=DB+DBC*U+DBS*V;
    END;

   PROCEDURE PERTVEN;(* кеплеровские члены и возмущения,         *)
                     (* вызванные Венерой                        *)
    VAR I: INTEGER;
    BEGIN
  C[0]:=1.0;  S[0]:=0.0;      C[-1]:=COS(M2);     S[-1]:=-SIN(M2);
FOR  I:=-1  DOWNTO  -4  DO  ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
TERM(1,  0,0,   259.74,  84547.39, -78342.34,  0.01,11683.22, 21203.79);
TERM(1,  0,1,     2.30,      5.04,     -7.52,  0.02,  138.55,   -71.01);
TERM(1,  0,2,     0.01,     -0.01,      0.01,  0.01,   -0.19,    -0.54);
TERM(2,  0,0,  -549.71,  10394.44,  -7955.45,  0.00, 2390.29,  4306.79);
TERM(2,  0,1,    -4.77,      8.97,     -1.53,  0.00,   28.49,   -14.18);
TERM(2,  0,2,     0.00,      0.00,      0.00,  0.00,   -0.04,    -0.11);
TERM(3,  0,0,  -234.04,   1748.74,  -1212.86,  0.00,  535.41,   984.33);
TERM(3,  0,1,    -2.03,      3.48,     -0.35,  0.00,    6.56,    -2.91);
TERM(4,  0,0,   -77.64,    332.63,   -219.23,  0.00,  124.40,   237.03);
TERM(4,  0,1,    -0.70,      1.10,     -0.08,  0.00,    1.59,    -0.59);
TERM(5,  0,0,   -23.59,     67.28,    -43.54,  0.00,   29.44,    58.77);
TERM(5,  0,1,    -0.23,      0.32,     -0.02,  0.00,    0.39,    -0.11);
TERM(6,  0,0,    -6.86,     14.06,     -9.18,  0.00,    7.03,    14.84);
TERM(6,  0,1,    -0.07,      0.09,     -0.01,  0.00,    0.10,    -0.02);
TERM(7,  0,0,    -1.94,      2.98,     -2.02,  0.00,    1.69,     3.80);
TERM(8,  0,0,    -0.54,      0.63,     -0.46,  0.00,    0.41,     0.98);
TERM(9,  0,0,    -0.15,      0.13,     -0.11,  0.00,    0.10,     0.25);
TERM(-1,-2,0,    -0.17,     -0.06,     -0.05,  0.14,   -0.06,    -0.07);
TERM(0, -1,0,     0.24,     -0.16,     -0.11, -0.16,    0.04,    -0.01);
TERM(0, -2,0,    -0.68,     -0.25,      0.26,  0.73,   -0.16,    -0.18);
TERM(0, -5,0,     0.37,      0.08,      0.06, -0.28,    0.13,     0.12);
TERM(1, -1,0,     0.58,     -0.41,      0.26,  0.36,    0.01,    -0.01);
TERM(1, -2,0,    -3.51,     -1.23,      0.23, -0.63,   -0.05,    -0.06);
```

```
 TERM( 1,-3,0,   0.08,    0.53,   -0.11,  0.04,   0.02,  -0.09);
 TERM( 1,-5,0,   1.44,    0.31,    0.30, -1.39,   0.34,   0.29);
 TERM( 2,-1,0,   0.15,   -0.11,    0.09,  0.12,   0.02,  -0.04);
 TERM( 2,-2,0,  -1.99,   -0.68,    0.65, -1.91,  -0.20,   0.03);
 TERM( 2,-3,0,  -0.34,   -1.28,    0.97, -0.26,   0.03,   0.03);
 TERM( 2,-4,0,  -0.33,    0.35,   -0.13, -0.13,  -0.01,   0.00);
 TERM( 2,-5,0,   7.19,    1.56,   -0.05,  0.12,   0.06,   0.05);
 TERM( 3,-2,0,  -0.52,   -0.18,    0.13, -0.39,  -0.16,   0.03);
 TERM( 3,-3,0,  -0.11,   -0.42,    0.36, -0.10,  -0.05,  -0.05);
 TERM( 3,-4,0,  -0.19,    0.22,   -0.23, -0.20,  -0.01,   0.02);
 TERM( 3,-5,0,   2.77,    0.49,   -0.45,  2.56,   0.40,  -0.12);
 TERM( 4,-5,0,   0.67,    0.12,   -0.09,  0.47,   0.24,  -0.08);
 TERM( 5,-5,0,   0.18,    0.03,   -0.02,  0.12,   0.09,  -0.03);
    END;
  PROCEDURE PERTEAR;(* возмущения, вызванные Землей          *)
    VAR I: INTEGER;
    BEGIN
      C[-1]:=COS(M3);     S[-1]:=-SIN(M3);
FOR  I:=-1  DOWNTO  -3  DO  ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
 TERM( 0,-4,0,  -0.11,   -0.07,   -0.08,  0.11,  -0.02,  -0.04);
 TERM( 1,-1,0,   0.10,   -0.20,    0.15,  0.07,   0.00,   0.00);
 TERM( 1,-2,0,  -0.35,    0.28,   -0.13, -0.17,  -0.01,   0.00);
 TERM( 1,-4,0,  -0.67,   -0.45,    0.00,  0.01,  -0.01,  -0.01);
 TERM( 2,-2,0,  -0.20,    0.16,    0.16, -0.20,  -0.01,   0.02);
 TERM( 2,-3,0,   0.13,   -0.02,    0.02,  0.14,   0.01,   0.00);
 TERM( 2,-4,0,  -0.33,   -0.18,    0.17, -0.31,  -0.04,   0.00);
    END;

  PROCEDURE PERTJUP;(* возмущения, вызванные Юпитером        *)
    VAR I: INTEGER;
    BEGIN
      C[-1]:=COS(M5);     S[-1]:=-SIN(M5);
FOR I:=-1 DOWNTO -2 DO ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
 TERM(-1,-1,0,  -0.08,    0.16,    0.15,  0.08,  -0.04,   0.01);
 TERM(-1,-2,0,   0.10,   -0.06,   -0.07, -0.12,   0.07,  -0.01);
 TERM( 0,-1,0,  -0.31,    0.48,   -0.02,  0.13,  -0.03,  -0.02);
 TERM( 0,-2,0,   0.42,   -0.26,   -0.38, -0.50,   0.20,  -0.03);
 TERM( 1,-1,0,  -0.70,    0.01,   -0.02, -0.63,   0.00,   0.03);
 TERM( 1,-2,0,   2.61,   -1.97,    1.74,  2.32,   0.01,   0.01);
 TERM( 1,-3,0,   0.32,   -0.15,    0.13,  0.28,   0.00,   0.00);
 TERM( 2,-1,0,  -0.18,    0.01,    0.00, -0.13,  -0.03,   0.03);
 TERM( 2,-2,0,   0.75,   -0.56,    0.45,  0.60,   0.08,  -0.17);
 TERM( 3,-2,0,   0.20,   -0.15,    0.10,  0.14,   0.04,  -0.08);
   END;

  PROCEDURE PERTSAT; (* возмущения, вызванные Сатурном      *)
```

```
    BEGIN
       C[-2]:=COS(2*M6);      S[-2]:=-SIN(2*M6);
   TERM( 1,-2,0,   -0.19,       0.33,         0.00,  0.00,       0.00,       0.00);
    END;

  BEGIN  (* MER200          *)

    DL:=0.0;   DR:=0.0;   DB:=0.0;
    M1:=P2*FRAC(0.4855407+415.2014314*T);
    M2:=P2*FRAC(0.1394222+162.5490444*T);
    M3:=P2*FRAC(0.9937861+  99.9978139*T);
    M5:=P2*FRAC(0.0558417+    8.4298417*T);
    M6:=P2*FRAC(0.8823333+    3.3943333*T);
    C1[0]:=1.0;        S1[0]:=0.0;
    C1[1]:=COS(M1);        S1[1]:=SIN(M1);        C1[-1]:=C1[1];        S1[-1]:=-S1[1];
  FOR   I:=2    TO    9    DO    ADDTHE(C1[I-1],S1[I-1],C1[1],S1[1],C1[I],S1[I]);
    PERTVEN; PERTEAR; PERTJUP; PERTSAT;
    DL :=   DL   +   (2.8+3.2*T);
     L := 360.0*FRAC(0.2151379 + M1/P2 + ((5601.7+1.1*T)*T
                                              +DL)/1296.0E3   );
      R:= 0.3952829     + 0.0000016*T         +DR*1.0E-6;
      B:= ( -2522.15     + (-30 18  + 0.04*T)  *  T    +    DB  )  /  3600.0;

  END;    (* MER200 *)
(*-------------------------------------------------------------------------*)
```

Венера

```
(*-------------------------------------------------------------------------*)
(* VEN200:  Венера; эклиптические координаты L,B,R (град. и а.е.)         *)
(*          равноденствие даты                                             *)
(*          (T: время в юлианских столетиях с J2000)                       *)
(           (    = (JED-2451545.0)/36525       )                           *)
(*-------------------------------------------------------------------------*)
PROCEDURE VEN200(T:REAL;VAR L,B,R:REAL);
  CONST  P2=6.283185307;
  VAR    C2,S2                :ARRAY [ 0..8] OF  REAL;
         C,S                  :ARRAY [-8..0] OF  REAL;
         M1,M2,M3,M4,M5,M6:REAL;
         U,V,  DL,DR,DB       :REAL;
         I                    :INTEGER;

  FUNCTION FRAC(X:REAL):REAL;
    BEGIN  X:=X-TRUNC(X);  IF  (X<0)  THEN  X:=X+1.0;  FRAC:=X   END;

  PROCEDURE ADDTHE(C1,S1,C2,S2:REAL; VAR C,S:REAL);
    BEGIN  C:=C1*C2-S1*S2;    S:=S1*C2+C1*S2;   END;
```

```
  PROCEDURE TERM(I1,I,IT:INTEGER;DLC,DLS,DRC,DRS,DBC,DBS:REAL);
    BEGIN
      IF   IT=0   THEN   ADDTHE(C2[I1],S2[I1],C[I],S[I],U,V)
                 ELSE BEGIN U:=U*T; V:=V*T END;
      DL:=DL+DLC*U+DLS*V; DR:=DR+DRC*U+DRS*V;
                                              DB:=DB+DBC*U+DBS*
    END;
  PROCEDURE PERTMER; (* возмущения, вызванные Меркурием
    BEGIN
      C[0]:=1.0;    S[0]:=0.0;    C[-1]:=COS(M1);    S[-1]:=-SIN(M1);
      ADDTHE(C[-1],S[-1],C[-1],S[-1],C[-2],S[-2]);
 TERM( 1,-1,0,    0.00,      0.00,        0.06, -0.09,      0.01,      0.00);
 TERM( 2,-1,0,    0.25,     -0.09,       -0.09, -0.27,      0.00,      0.00);
 TERM( 4,-2,0,   -0.07,     -0.08,       -0.14,  0.14,     -0.01,     -0.01);
 TERM( 5,-2,0,   -0.35,      0.08,        0.02,  0.09,      0.00,      0.00);
     END;

  PROCEDURE PERTEAR;(* кеплеровские члены и возмущения,               *)
                    (* вызванные Землей                               *)
    VAR I: INTEGER;
    BEGIN
      C[-1]·=COS(M3);      S[-1]:=-SIN(M3);
FOR  I:=-1  DOWNTO  -7  DO  ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
TERM( 1,  0,0,    2.37,   2793.23,  -4899.07,  0.11,  9995.27,  7027.22);
TERM( 1,  0,1,    0.10,    -19.65,     34.40,  0.22,    64.95,   -86.10);
TERM( 1,  0,2,    0.06,      0.04,     -0.07,  0.11,    -0.55,    -0.07);
TERM( 2,  0,0, -170.42,     73.13,    -16.59,  0.00,    67.71,    47.56),
TERM( 2,  0,1,    0.93,      2.91,      0.23,  0.00,    -0.03,    -0.92);
TERM( 3,  0,0,   -2.31,      0.90,     -0.08,  0.00,     0.04,     2.09),
TERM( 1, -1,0,   -2.38,     -4.27,      3.27, -1.82,     0.00,     0.00);
TERM( 1, -2,0,    0.09,      0.00,     -0.08,  0.05,    -0.02,    -0.25),
TERM( 2, -2,0,   -9.57,     -5.93,      8.57,-13.83,    -0.01,    -0.01);
TERM( 2, -3,0,   -2.47,     -2.40,      0.83, -0.95,     0.16,     0.24);
TERM( 3, -2,0,   -0.09,     -0.05,      0.08, -0.13,    -0.28,     0.12),
TERM( 3, -3,0,    7.12,      0.32,     -0.62, 13.76,    -0.07,     0.01);
TERM( 3, -4,0,   -0.65,     -0.17,      0.18, -0.73,     0.10,     0.05);
TERM( 3, -5,0,   -1.08,     -0.95,     -0.17,  0.22,    -0.03,    -0.03);
TERM( 4, -3,0,    0.06,      0.00,     -0.01,  0.08,     0.14,    -0.18);
TERM( 4, -4,0,    0.93,     -0.46,      1.06,  2.13,    -0.01,     0.01);
TERM( 4, -5,0,   -1.53,      0.38,     -0.64, -2.54,     0.27,     0.00);
TERM( 4, -6,0,   -0.17,     -0.05,      0.03, -0.11,     0.02,     0.00);
TERM( 5, -5,0,    0.18,     -0.28,      0.71,  0.47,    -0.02,     0.04);
TERM( 5, -6,0,    0.15,     -0.14,      0.30,  0.31,    -0.04,     0.03);
TERM( 5, -7,0,   -0.08,      0.02,     -0.03, -0.11,     0.01,     0.00);
TERM( 5, -8,0,   -0.23,      0.00,      0.01, -0.04,     0.00,     0.00);
TERM( 6, -6,0,    0.01,     -0.14,      0.39,  0.04,     0.00,    -0.01);
TERM( 6, -7,0,    0.02,     -0.05,      0.12,  0.04,    -0.01,     0.01);
```

```
TERM( 6,-8,0,    0.10,    -0.10,    0.19,  0.19,   -0.02,   0.02);
TERM( 7,-7,0,   -0.03,    -0.06,    0.18, -0.08,    0.00,   0.00);
TERM( 8,-8,0,   -0.03,    -0.02,    0.06, -0.08,    0.00,   0.00);
    END;

  PROCEDURE PERTMAR; (* возмущения, вызванные Марсом           *)
    VAR I: INTEGER;
    BEGIN
      C[-1]:=COS(M4);     S[-1]:=-SIN(M4);
FOR  I:=-1  DOWNTO  -2  DO  ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
TERM( 1,-3,0,   -0.65,     1.02,   -0.04, -0.02,   -0.02,   0.00);
TERM( 2,-2,0,   -0.05,     0.04,   -0.09, -0.10,    0.00,   0.00);
TERM( 2,-3,0,   -0.50,     0.45,   -0.79, -0.89,    0.01,   0.03);
     END;

  PROCEDURE PERTJUP;(* возмущения, вызванные Юпитером          *)
    VAR I: INTEGER;
    BEGIN
      C[-1]:=COS(M5);     S[-1]:=-SIN(M5);
FOR  I:=-1  DOWNTO  -2  DO  ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
TERM( 0,-1,0,   -0.05,     1.56,    0.16,  0.04,   -0.08,  -0.04);
TERM( 1,-1,0,   -2.62,     1.40,   -2.35, -4.40,    0.02,   0.03);
TERM( 1,-2,0,   -0.47,    -0.08,    0.12, -0.76,    0.04,  -0.18);
TERM( 2,-2,0,   -0.73,    -0.51,    1.27, -1.82,   -0.01,   0.01);
TERM( 2,-3,0,   -0.14,    -0.10,    0.25, -0.34,    0.00,   0.00);
TERM( 3,-3,0,   -0.01,     0.04,   -0.11, -0.02,    0.00,   0.00);
     END;

  PROCEDURE PERTSAT;(* возмущения, вызванные Сатурном          *)
     BEGIN
      C[-1]:=COS(M6);     S[-1]:=-SIN(M6);
TERM( 0,-1,0,    0.00,     0.21,    0.00,  0.00,    0.00,  -0.01);
TERM( 1,-1,0,   -0.11,    -0.14,    0.24, -0.20,    0.01,   0.00);
    END;

  BEGIN (* VEN200     *)

    DL:=0.0;   DR:=0.0;   DB:=0.0;
    M1:=P2*FRAC(0.4861431+415.2018375*T);
    M2:=P2*FRAC(0.1400197+162.5494552*T);
    M3:=P2*FRAC(0.9944153+  99.9982208*T);
    M4:=P2*FRAC(0.0556297+  53.1674631*T);
    M5:=P2*FRAC(0.0567028+   8.4305083*T);
    M6:=P2*FRAC(0.8830539+   3.3947206*T);

 C2[0]:=1.0;    S2[0]:=0.0;    C2[1]:=COS(M2);    S2[1]:=SIN(M2);
 FOR I:=2 TO 8 DO ADDTHE(C2[I-1],S2[I-1],C2[1],S2[1],C2[I],S2[I]);
```

```
  PERTMER; PERTEAR; PERTMAR; PERTJUP; PERTSAT;
  DL:=DL + 2.74*SIN(P2*(0.0764+0.4174*T))
        +      0.27*SIN(P2*(0.9201+0.3307*T));
  DL:= DL + (1.9+1.8*T);
  L := 360.0*FRAC(0.3654783 + M2/P2  + ((5071.2+1.1*T)*T
                                      + DL)/1296.0E3  );
  R := 0.7233482 - 0.0000002*T    +   DR*1.0E-6;
  B := ( -67.70 + ( 0.04 + 0.01*T) * T      +   DB ) / 3600.0;

END;   (* VEN200  *)
(*-----------------------------------------------------------*)
```

Земля

Для вычисления гелиоцентрических координат Земли нет специальной программы. Координаты Земли в любой момент можно определить из гелиоцентрических координат Солнца, получаемых по подпрограмме SUN200 (гл.2). Для этого меняют знак эклиптической широты Солнца и добавляют 180° к эклиптической долготе.

Марс

```
(*-----------------------------------------------------------*)
(* MAR200:  Марс; эклиптические координаты L,B,R (град. и а.е.)  *)
(*          равноденствие даты                                  *)
(*          (T· время в юлианских столетиях с J2000)            *)
(*          (    = (JED-2451545.0)/36525       )                *)
(*-----------------------------------------------------------*)
PROCEDURE  MAR200(T:REAL;VAR  L,B,R:REAL);


  CONST  P2=6 283185307;
   VAR C4,S4             :  ARRAY [-2..16] OF REAL,
       C,S               :  ARRAY [-9.. 0] OF REAL;
       M2,M3,M4,M5,M6:REAL;
       U,V, DL,DR,DB :  REAL;
       I                 :  INTEGER;

   FUNCTION FRAC(X REAL):REAL;
     BEGIN  X:=X-TRUNC(X); IF (X<0) THEN X:=X+1.0; FRAC:=X  END;

   PROCEDURE ADDTHE(C1,S1,C2,S2:REAL; VAR C,S:REAL);
     BEGIN   C:=C1*C2-S1*S2;  S:=S1*C2+C1*S2;  END;

   PROCEDURE TERM(I1,I,IT:INTEGER;DLC,DLS,DRC,DRS,DBC,DBS:REAL);
     BEGIN
```

```
      IF   IT=0   THEN   ADDTHE(C4[I1],S4[I1],C[I],S[I],U,V)
                  ELSE BEGIN U:=U*T; V:=V*T END;
      DL:=DL+DLC*U+DLS*V; DR:=DR+DRC*U+DRS*V; DB:=DB+DBC*U+DBS*V;
    END;

  PROCEDURE PERTVEN; (* возмущения, вызванные Венерой          *)
    BEGIN
      C[0]:=1.0;   S[0]:=0.0;   C[-1]:=COS(M2);   S[-1]:=-SIN(M2);
      ADDTHE(C[-1],S[-1],C[-1],S[-1],C[-2],S[-2]);
TERM( 0,-1,0,   -0.01,     -0.03,    0.10,   -0.04,      0.00,    0.00);
TERM( 1,-1,0,    0.05,      0.10,   -2.08,    0.75,      0.00,    0.00);
TERM( 2,-1,0,   -0 25,     -0.57,   -2.58,    1.18,      0.05,   -0.04);
TERM( 2,-2,0,    0.02,      0.02,    0.13,   -0.14,      0.00,    0.00);
TERM( 3,-1,0,    3.41,      5.38,    1.87,   -1.15,      0.01,   -0.01);
TERM( 3,-2,0,    0.02,      0.02,    0.11,   -0.13,      0.00,    0.00);
TERM( 4,-1,0,    0.32,      0.49,   -1.88,    1.21,     -0.07,    0.07);
TERM( 4,-2,0,    0.03,      0.03,    0.12   -0.14,      0.00,    0.00);
TERM( 5,-1,0,    0.04,      0.06,   -0.17,    0.11,     -0.01,    0.01);
TERM( 5,-2,0,    0.11,      0.09,    0.35,   -0.43,     -0.01,    0.01);
TERM( 6,-2,0,   -0.36,     -0.28,   -0.20,    0.25,      0.00,    0.00);
TERM( 7,-2,0,    0.03,     -0.03,    0.11,   -0.13,      0.00,   -0.01);
    END;

  PROCEDURE PERTEAR;(* кеплеровские члены и возмущения           *)
                    (* вызванные Землей                          *)
    VAR I: INTEGER;
    BEGIN
      C[-1]:=COS(M3);        S[-1]:=-SIN(M3);
FOR  I:=-1  DOWNTO  -8  DO  ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
TERM( 1,  0,0,  -5.32,  38481.97,-141856.04,   0.40,  -6321.67, 1876.89);
TERM( 1,  0,1,  -1.12,     37.98,   -138.67,  -2.93,     37.28,  117.48);
TERM( 1,  0,2,  -0.32,     -0.03,      0.12,  -1.19,      1.04,   -0.40);
TERM( 2,  0,0,  28.28,   2285.80,  -6608.37,   0.00,   -589.35,  174.81);
TERM( 2,  0,1,   1.64,      3.37,    -12.93,   0.00,      2.89,   11.10);
TERM( 2,  0,2,   0.00,      0.00,      0.00,   0.00,      0.10,   -0.03);
TERM( 3,  0,0,   5.31,    189.29,   -461.81,   0.00,    -61.98,   18.53);
TERM( 3,  0,1,   0.31,      0.35,     -1.36,   0.00,      0.25,    1.19);
TERM( 4,  0,0,   0.81,     17.96,    -38.26,   0.00,     -6.88,    2.08);
TERM( 4,  0,1,   0.05,      0.04,     -0.15,   0.00,      0.02,    0.14);
TERM( 5,  0,0,   0.11,      1.83,     -3.48,   0.00,     -0.79,    0.24);
TERM( 6,  0,0,   0.02,      0.20,     -0.34,   0.00,     -0.09,    0.03);
TERM(-1,-1,0,    0.09,      0.06,      0.14,  -0.22,      0.02,   -0.02);
TERM( 0,-1,0,    0.72,      0.49,      1.55,  -2.31,      0.12,   -0.10);
TERM( 1,-1,0,    7.00,      4.92,     13.93, -20.48,      0.08,   -0.13);
TERM( 2,-1,0,   13.08,    4.89,       -4.53,  10.01,     -0.05,    0.13);
TERM( 2,-2,0,    0.14,    0.05,       -0.48,  -2.66,      0.01,    0.14);
TERM( 3,-1,0,    1.38,    0.56,       -2.00,   4.85,     -0.01,    0.19);
```

```
TERM( 3,-2,0,   -6.85,   2.68,    8.38,  21.42,   0.00,   0.03);
TERM( 3,-3,0,   -0.08,   0.20,    1.20,   0.46,   0.00,   0.00);
TERM( 4,-1,0,    0.16,   0.07,   -0.19,   0.47,  -0.01,   0.05);
TERM( 4,-2,0,   -4.41,   2.14,   -3.33,  -7.21,  -0.07,  -0.09);
TERM( 4,-3,0,   -0.12,   0.33,    2.22,   0.72,  -0.03,  -0.02);
TERM( 4,-4,0,   -0.04,  -0.06,   -0.36,   0.23,   0.00,   0.00);
TERM( 5,-2,0,   -0.44,   0.21,   -0.70,  -1.46,  -0.06,  -0.07);
TERM( 5,-3,0,    0.48,  -2.60,   -7.25,  -1.37,   0.00,   0.00);
TERM( 5,-4,0,   -0.09,  -0.12,   -0.66,   0.50,   0.00,   0.00);
TERM( 5,-5,0,    0.03,   0.00,    0.01,  -0.17,   0.00,   0.00);
TERM( 6,-2,0,   -0.05,   0.03,   -0.07,  -0.15,  -0.01,  -0.01);
TERM( 6,-3,0,    0.10,  -0.96,    2.36,   0.30,   0.04,   0.00);
TERM( 6,-4,0,   -0.17,  -0.20,   -1.09,   0.94,   0.02,  -0.02);
TERM( 6,-5,0,    0.05,   0.00,    0.00,  -0.30,   0.00,   0.00);
TERM( 7,-3,0,    0.01,  -0.10,    0.32,   0.04,   0.02,   0.00);
TERM( 7,-4,0,    0.86,   0.77,    1.86,  -2.01,   0.01,  -0.01);
TERM( 7,-5,0,    0.09,  -0.01,   -0.05,  -0.44,   0.00,   0.00);
TERM( 7,-6,0,   -0.01,   0.02,    0.10,   0.08,   0.00,   0.00);
TERM( 8,-4,0,    0.20,   0.16,   -0.53,   0.64,  -0.01,   0.02);
TERM( 8,-5,0,    0.17,  -0.03,   -0.14,  -0.84,   0.00,   0.01);
TERM( 8,-6,0,   -0.02,   0.03,    0.16,   0.09,   0.00,   0.00);
TERM( 9,-5,0,   -0.55,   0.15,    0.30,   1.10,   0.00,   0.00);
TERM( 9,-6,0,    0.02,   0.04,    0.20,   0.10,   0.00,   0.00);
TERM(10,-5,0,   -0.09,   0.03,   -0.10,  -0.33,   0.00,  -0.01);
TERM(10,-6,0,   -0.05,   0.11,    0.48,   0.21,  -0.01,   0.00);
TERM(11,-6,0,    0.10,  -0.35,   -0.52,  -0.15,   0.00,   0.00);
TERM(11,-7,0,   -0.01,  -0.02,   -0.10,   0.07,   0.00,   0.00);
TERM(12,-6,0,    0.01,  -0.04,    0.18,   0.04,   0.01,   0.00);
TERM(12,-7,0,   -0.05,  -0.07,   -0.29,   0.20,   0.01,   0.00);
TERM(13,-7,0,    0.23,   0.27,    0.25,  -0.21,   0.00,   0.00);
TERM(14,-7,0,    0.02,   0.03,   -0.10,   0.09,   0.00,   0.00);
TERM(14,-8,0,    0.05,   0.01,    0.03,  -0.23,   0.00,   0.03);
TERM(15,-8,0,   -1.53,   0.27,    0.06,   0.42,   0.00,   0.00);
TERM(16,-8,0,   -0.14,   0.02,   -0.10,  -0.55,  -0.01,  -0.02);
TERM(16,-9,0,    0.03,  -0.06,   -0.25,  -0.11,   0.00,   0.00);
   END;

 PROCEDURE PERTJUP; (* возмущения, вызванные Юпитером        *)
   VAR I: INTEGER;
   BEGIN
     C[-1]:=COS(M5);     S[-1]:=-SIN(M5);
FOR I:=-1 DOWNTO -4 DO ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
TERM(-2,-1,0,    0.05,   0.03,    0.08,  -0.14,   0.01,  -0.01);
TERM(-1,-1,0,    0.39,   0.27,    0.92,  -1.50,  -0.03,  -0.06);
TERM(-1,-2,0,   -0.16,   0.03,    0.13,   0.67,  -0.01,   0.06);
TERM(-1,-3,0,   -0.02,   0.01,    0.05,   0.09,   0.00,   0.01);
```

```
TERM( 0,-1,0,    3.56,    1.13,    -5.41,   -7.18,     -0.25,  -0.24);
TERM( 0,-2,0,   -1.44,    0.25,     1.24,    7.96,      0.02,   0.31);
TERM( 0,-3,0,   -0.21,    0.11,     0.55,    1.04,      0.01,   0.05);
TERM( 0,-4,0,   -0.02,    0.02,     0.11,    0.11,      0.00,   0.01);
TERM( 1,-1,0,   16.67,  -19.15,    61.00,   53.36,     -0.06,  -0.07);
TERM( 1,-2,0,  -21.64,    3.18,    -7.77,  -54.64,     -0.31,   0.50);
TERM( 1,-3,0,   -2.82,    1.45,    -2.53,   -5.73,      0.01,   0.07);
TERM( 1,-4,0,   -0.31,    0.28,    -0.34,   -0.51,      0.00,   0.00);
TERM( 2,-1,0,    2.15,   -2.29,     7.04,    6.94,      0.33,   0.19);
TERM( 2,-2,0,  -15.69,    3.31,   -15.70,  -73.17,     -0.17,  -0.25);
TERM( 2,-3,0,   -1.73,    1.95,    -9.19,   -7.20,      0.02,  -0.03);
TERM( 2,-4,0,   -0.01,    0.33,    -1.42,    0.08,      0.01,  -0.01);
TERM( 2,-5,0,    0.03,    0.03,    -0.13,    0.12,      0.00,   0.00);
TERM( 3,-1,0,    0.26,   -0.28,     0.73,    0.71,      0.08,   0.04);
TERM( 3,-2,0,   -2.06,    0.46,    -1.61,   -6.72,     -0.13,  -0.25);
TERM( 3,-3,0,   -1.28,   -0.27,     2.21,   -6.90,     -0.04,  -0.02);
TERM( 3,-4,0,   -0.22,    0.08,    -0.44,   -1.25,      0.00,   0.01);
TERM( 3,-5,0,   -0.02,    0.03,    -0.15,   -0.08,      0.00,   0.00);
TERM( 4,-1,0,    0.03,   -0.03,     0.08,    0.08,      0.01,   0.01);
TERM( 4,-2,0,   -0.26,    0.06,    -0.17,   -0.70,     -0.03,  -0.05);
TERM( 4,-3,0,   -0.20,   -0.05,     0.22,   -0.79,     -0.01,  -0.02);
TERM( 4,-4,0,   -0.11,   -0.14,     0.93,   -0.60,      0.00,   0.00);
TERM( 4,-5,0,   -0.04,   -0.02,     0.09,   -0.23,      0.00,   0.00);
TERM( 5,-4,0,   -0.02,   -0.03,     0.13,   -0.09,      0.00,   0.00);
TERM( 5,-5,0,    0.00,   -0.03,     0.21,    0.01,      0.00,   0.00);
  END;

 PROCEDURE PERTSAT;(* возмущения, вызванные Сатурном            *)
  VAR I: INTEGER;
  BEGIN
    C[-1]:=COS(M6); S[-1]:=-SIN(M6);
FOR I:=-1 DOWNTO -3 DO ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
TERM(-1,-1,0,    0.03,    0.13,     0.48,   -0.13,      0.02,   0.00);
TERM( 0,-1,0,    0.27,    0.84,     0.40,   -0.43,      0.01,  -0.01);
TERM( 0,-2,0,    0.12,   -0.04,    -0.33,   -0.55,     -0.01,  -0.02);
TERM( 0,-3,0,    0.02,   -0.01,    -0.07,   -0.08,      0.00,   0.00);
TERM( 1,-1,0,    1.12,    0.76,    -2.66,    3.91,     -0.01,   0.01);
TERM( 1,-2,0,    1.49,   -0.95,     3.07,    4.83,      0.04,  -0.05);
TERM( 1,-3,0,    0.21,   -0.18,     0.55,    0.64,      0.00,   0.00);
TERM( 2,-1,0,    0.12,    0.10,    -0.29,    0.34,     -0.01,   0.02);
TERM( 2,-2,0,    0.51,   -0.36,     1.61,    2.25,      0.03,   0.01);
TERM( 2,-3,0,    0.10,   -0.10,     0.50,    0.43,      0.00,   0.00);
TERM( 2,-4,0,    0.01,   -0.02,     0.11,    0.05,      0.00,   0.00);
TERM( 3,-2,0,    0.07,   -0.05,     0.16,    0.22,      0.01,   0.01);
  END;

 BEGIN (* MAR200       *)
```

```
    DL:=0.0;   DR:=0.0;   DB:=0.0;
    M2:=P2*FRAC(0.1382208+162.5482542*T);
    M3:=P2*FRAC(0.9926208+ 99.9970236*T);
    M4:=P2*FRAC(0.0538553+ 53.1662736*T);
    M5:=P2*FRAC(0.0548944+  8.4290611*T);
    M6:=P2*FRAC(0.8811167+  3.3935250*T);
    C4[0]:=1.0;   S4[0]:=0.0;     C4[1]:=COS(M4);    S4[1]:=SIN(M4);
  FOR  I:=2  TO  16  DO  ADDTHE(C4[I-1],S4[I-1],C4[1],S4[1],C4[I],S4[I]);
     FOR I:=-2 TO -1 DO BEGIN C4[I]:=C4[-I]; S4[I]:=-S4[-I] END;
    PERTVEN; PERTEAR; PERTJUP; PERTSAT;
    DL:=DL + 52.49*SIN(P2*(0.1868+0.0549*T))
           +  0.61*SIN(P2*(0.9220+0.3307*T))
           +  0.32*SIN(P2*(0.4731+2.1485*T))
           +  0.28*SIN(P2*(0.9467+0.1133*T));
   DL :=DL +      (0.14+0.87*T-0.11*T*T);
   L  :=360.0*FRAC(0.9334591+M4/P2+((6615.5+1.1*T)*T+DL)/1296.0E3);
   R  :=   1.5303352  + 0.0000131*T    +    DR*1.0E-6;
   B  := ( 596.32     + (-2.92 - 0.10*T) * T   +   DB ) / 3600.0;
  END;
(*-------------------------------------------------------------*)
```

Юпитер

Полный текст программы JUP200 с пояснениями дан в гл.5.

Сатурн

```
(*-------------------------------------------------------------*)
(* SAT200:  Сатурн; эклиптические координаты L,B,R (град. и а.е.) *)
(*          равноденствие даты                                  *)
(*          (T: время в юлианских столетиях с J2000)            *)
(*          (    = (JED-2451545.0)/36525          )             *)
(*-------------------------------------------------------------*)
PROCEDURE SAT200(T:REAL;VAR L,B,R:REAL);

  CONST P2=6.283185307;

   VAR C6,S6          : ARRAY [ 0..11] OF REAL;
       C,S            : ARRAY [-6.. 1] OF REAL;
       M5,M6,M7,M8 : REAL;
       U,V, DL,DR,DB: REAL;
       I              : INTEGER;

   FUNCTION FRAC(X:REAL):REAL;
     BEGIN X:=X-TRUNC(X); IF (X<0) THEN X:=X+1.0; FRAC:=X  END;

   PROCEDURE ADDTHE(C1,S1,C2,S2:REAL; VAR C,S:REAL);
```

```
    BEGIN  C:=C1*C2-S1*S2;  S:=S1*C2+C1*S2;  END;

  PROCEDURE TERM(I6,I,IT:INTEGER;DLC,DLS,DRC,DRS,DBC,DBS:REAL);
   BEGIN
     IF  IT=0  THEN  ADDTHE(C6[I6],S6[I6],C[I],S[I],U,V)
               ELSE BEGIN U:=U*T; V:=V*T END;
     DL:=DL+DLC*U+DLS*V; DR:=DR+DRC*U+DRS*V; DB:=DB+DBC*U+DBS*V;
    END;

  PROCEDURE PERTJUP;(* кеплеровские члены и возмущения,        *)
                    (* вызванные Юпитером                       *)
    VAR I: INTEGER;
    BEGIN
     C[0]:=1.0;    S[0]:=0.0;    C[1]:=COS(M5);     S[1]:=SIN(M5);
FOR I:=0 DOWNTO -5 DO ADDTHE(C[I],S[I],C[1],-S[1],C[I-1],S[I-1]);
TERM(0, -1,0,     12.0,     -1.4,     -13.9,     6.4,     1.2,   -1.8);
TERM(0, -2,0,      0.0,     -0.2,      -0.9,     1.0,     0.0,   -0.1);
TERM(1,  1,0,      0.9,      0.4,      -1.8,     1.9,     0.2,    0.2);
TERM(1,  0,0,   -348.3,  22907.7,  -52915.5,  -752.2, -3266.5, 8314.4);
TERM(1,  0,1,   -225.2,   -146.2,     337.7,   521.3,    79.6,   17.4);
TERM(1,  0,2,      1.3,     -1.4,       3.2,     2.9,     0.1,   -0.4);
TERM(1, -1,0,     -1.0,    -30.7,     108.6,  -815.0,    -3.6,   -9.3);
TERM(1, -2,0,     -2.0,     -2.7,      -2.1,   -11.9,    -0.1,   -0.4);
TERM(2,  1,0,      0.1,      0.2,      -1.0,     0.3,     0.0,    0.0);
TERM(2,  0,0,     44.2,    724.0,   -1464.3,   -34.7,  -188.7,  459.1);
TERM(2,  0,1,    -17.0,    -11.3,      18.9,   -28.6,     1.0,   -3.7);
TERM(2, -1,0,     -3.5,   -426.6,    -546.5,   -26.5,    -1.6,   -2.7);
TERM(2, -1,1,      3.5,     -2.2,      -2.6,    -4.3,     0.0,    0.0);
TERM(2, -2,0,     10.5,    -30.9,    -130.5,   -52.3,    -1.9,    0.2);
TERM(2, -3,0,     -0.2,     -0.4,      -1.2,    -0.1,    -0.1,    0.0);
TERM(3,  0,0,      6.5,     30.5,     -61.1,     0.4,   -11.6,   28.1);
TERM(3,  0,1,     -1.2,     -0.7,       1.1,    -1.8,    -0.2,   -0.6);
TERM(3, -1,0,     29.0,    -40.2,      98.2,    45.3,     3.2,   -9.4);
TERM(3, -1,1,      0.6,      0.6,      -1.0,     1.3,     0.0,    0.0);
TERM(3, -2,0,    -27.0,    -21.1,     -68.5,     8.1,   -19.8,    5.4);
TERM(3, -2,1,      0.9,     -0.5,      -0.4,    -2.0,    -0.1,   -0.8);
TERM(3, -3,0,     -5.4,     -4.1,     -19.1,    26.2,    -0.1,   -0.1);
TERM(4,  0,0,      0.6,      1.4,      -3.0,    -0.2,    -0.6,    1.6);
TERM(4, -1,0,      1.5,     -2.5,      12.4,     4.7,     1.0,   -1.1);
TERM(4, -2,0,   -821.9,     -9.6,     -26.0,  1873.6,   -70.5,   -4.4);
TERM(4, -2,1,      4.1,    -21.9,     -50.3,    -9.9,     0.7,   -3.0);
TERM(4, -3,0,     -2.0,     -4.7,     -19.3,     8.2,    -0.1,   -0.3);
TERM(4, -4,0,     -1.5,      1.3,       6.5,     7.3,     0.0,    0.0);
TERM(5, -2,0,  -2627.6,  -1277.3,     117.4,  -344.1,   -13.8,   -4.3);
TERM(5, -2,1,     63.0,    -98.6,      12.7,     6.7,     0.1,   -0.2);
TERM(5, -2,2,      1.7,      1.2,      -0.2,     0.3,     0.0;    0.0);
TERM(5, -3,0,      0.4,     -3.6,     -11.3,    -1.6,     0.0,   -0.3);
```

```
TERM( 5,-4,0,     -1.4,      0.3,       1.5,     6.3,     -0.1,    0.0);
TERM( 5,-5,0,      0.3,      0.6,       3.0,    -1.7,      0.0,    0.0);
TERM( 6,-2,0,   -146.7,    -73.7,     166.4,  -334.3,    -43.6,  -46.7);
TERM( 6,-2,1,      5.2,     -6.8,      15.1,    11.4,      1.7,   -1.0);
TERM( 6,-3,0,      1.5,     -2.9,      -2.2,    -1.3,      0.1,   -0.1);
TERM( 6,-4,0,     -0.7,     -0.2,      -0.7,     2.8,      0.0,    0.0);
TERM( 6,-5,0,      0.0,      0.5,       2.5,    -0.1,      0.0,    0.0);
TERM( 6,-6,0,      0.3,     -0.1,      -0.3,    -1.2,      0.0,    0.0);
TERM( 7,-2,0,     -9.6,     -3.9,       9.6,   -18.6,     -4.7,   -5.3);
TERM( 7,-2,1,      0.4,     -0.5,       1.0,     0.9,      0.3,   -0.1);
TERM( 7,-3,0,      3.0,      5.3,       7.5,    -3.5,      0.0,    0.0);
TERM( 7,-4,0,      0.2,      0.4,       1.6,    -1.3,      0.0,    0.0);
TERM( 7,-5,0,     -0.1,      0.2,       1.0,     0.5,      0.0,    0.0);
TERM( 7,-6,0,      0.2,      0.0,       0.2,    -1.0,      0.0,    0.0);
TERM( 8,-2,0,     -0.7,     -0.2,       0.6,    -1.2,     -0.4,   -0.4);
TERM( 8,-3,0,      0.5,      1.0,      -2.0,     1.5,      0.1,    0.2);
TERM( 8,-4,0,      0.4,      1.3,       3.6,    -0.9,      0.0,   -0.1);
TERM( 9,-4,0,      4.0,     -8.7,     -19.9,    -9.9,      0.2,   -0.4);
TERM( 9,-4,1,      0.5,      0.3,       0.8,    -1.8,      0.0,    0.0);
TERM(10,-4,0,     21 3,    -16.8,       3.3,     3.3,      0.2,   -0.2);
TERM(10,-4,1,      1.0,      1.7,      -0.4,     0.4,      0.0,    0.0);
TERM(11,-4,0,      1.6,     -1.3,       3.0,     3.7,      0.8,   -0.2);
 END;

  PROCEDURE PERTURA;(* возмущения, вызванные Ураном          *)
    VAR I: INTEGER;
    BEGIN
      C[-1]:=COS(M7);      S[-1]:=-SIN(M7);
FOR  I:=-1  DOWNTO  -4  DO  ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
      TERM( 0,-1,0,    1.0,     0.7,     0.4,    -1.5,    0.1,    0.0);
      TERM( 0,-2,0,    0.0,    -0.4,    -1.1,     0.1,   -0.1,   -0.1);
      TERM( 0,-3,0,   -0.9,    -1.2,    -2.7,     2.1,   -0.5,   -0.3);
      TERM( 1,-1,0,    7.8,    -1.5,     2.3,    12.7,    0.0,    0.0);
      TERM( 1,-2,0,   -1.1,    -8.1,     5.2,    -0.3,   -0.3,   -0.3);
      TERM( 1,-3,0,  -16.4,   -21.0,    -2.1,     0.0,    0.4,    0.0);
      TERM( 2,-1,0,    0.6,    -0.1,     0.1,     1.2,    0.1,    0.0);
      TERM( 2,-2,0,   -4.9,   -11.7,    31.5,   -13.3,    0.0,   -0.2);
      TERM( 2,-3,0,   19.1,    10.0,   -22.1,    42.1,    0.1,   -1.1);
      TERM( 2,-4,0,    0.9,    -0.1,     0.1,     1.4,    0.0,    0.0);
      TERM( 3,-2,0,   -0.4,    -0.9,     1.7,    -0.8,    0.0,   -0.3);
      TERM( 3,-3,0,    2.3,     0.0,     1.0,     5.7,    0.3,    0.3);
      TERM( 3,-4,0,    0.3,    -0.7,     2.0,     0.7,    0.0,    0.0);
      TERM( 3,-5,0,   -0.1,    -0.4,     1.1,    -0.3,    0.0,    0.0);
   END;

  PROCEDURE PERTNEP; (* возмущения, вызванные Нептуном        *)
   BEGIN
```

```
    C[-1]:=COS(M8);      S[-1]:=-SIN(M8);
      ADDTHE(C[-1],S[-1],C[-1],S[-1],C[-2],S[-2]);
      TERM( 1,-1,0,     -1.3,     -1.2,       2.3,     -2.5,     0.0,     0.0);
      TERM( 1,-2,0,      1.0,     -0.1,       0.1,      1.4,     0.0,     0.0);
      TERM( 2,-2,0,      1.1,     -0.1,       0.2,      3.3,     0.0,     0.0);
    END;

  PROCEDURE PERTJUR; (* возмущения, вызванные Юпитером и Ураном *)
    VAR PHI,X,Y: REAL;
    BEGIN
      PHI:=(-2*M5+5*M6-3*M7);  X:=COS(PHI);          Y :=SIN(PHI);
      DL :=DL-0.8*X-0.1*Y;  DR:=DR-0.2*X+1.8*Y;      DB:=DB+0.3*X+0.5*Y;
      ADDTHE(X,Y,C6[1],S6[1],X,Y);
      DL :=DL+(+2.4-0.7*T)*X+(27.8-0.4*T)*Y;         DR:=DR+2.1*X-0.2*Y;
      ADDTHE(X,Y,C6[1],S6[1],X,Y);
      DL :=DL+0.1*X+1.6*Y;  DR:=DR-3.6*X+0.3*Y;      DB:=DB-0.2*X+0.6*Y;
    END;

  BEGIN  (* SAT200          *)

    DL   :=0.0;   DR:=0.0;   DB.=0.0;
    M5   :=P2*FRAC(0.0565314+8 4302963*T);
    M6   :=P2*FRAC(0.8829867+3.3947688*T);
    M7   :=P2*FRAC(0.3969537+1 1902586*T);
    M8   :=P2*FRAC(0.7208473+0 6068623*T);
    C6[0]:=1.0;   S6[0]:=0.0;       C6[1]:=COS(M6);    S6[1]:=SIN(M6);
FOR  I.=2   TO   11   DO   ADDTHE(C6[I-1],S6[I-1],C6[1],S6[1],C6[I],S6[I]);
    PERTJUP; PERTURA; PERTNEP; PERTJUR;
    L:= 360.0*FRAC(0.2561136 + M6/P2
       +   ((5018.6+T*1.9)*T   +DL)/1296.0E3   );
    R:= 9.557584 - 0.000186*T    +    DR*1.0E-5;
    B:= ( 175.1 - 10.2*T + DB ) / 3600.0;

  END;  (* SAT200          *)
(*-----------------------------------------------------------------   *)
```

Уран

```
(*-----------------------------------------------------------------   *)
(* URA200:  Уран; эклиптические координаты L,B,R (град. и а.е.)       *)
(*          равноденствие даты                                         *)
(*          (T: время в юлианских столетиях с J2000)                  *)
(*          (   = (JED-2451545.0)/36525          )                    *)
(*-----------------------------------------------------------------   *)
PROCEDURE URA200(T:REAL;VAR L,B,R:REAL);

 CONST P2=6.283185307;
```

```
VAR C7,S7           : ARRAY [-2..7] OF REAL;
    C,S             : ARRAY [-8..0] OF REAL;
    M5,M6,M7,M8 : REAL;
    U,V, DL,DR,DB: REAL;
    I               : INTEGER;

FUNCTION FRAC(X:REAL):REAL;
  BEGIN X:=X-TRUNC(X); IF (X<0) THEN X:=X+1.0; FRAC:=X   END;

PROCEDURE ADDTHE(C1,S1,C2,S2:REAL; VAR C,S:REAL);
  BEGIN   C:=C1*C2-S1*S2; S:=S1*C2+C1*S2; END;

PROCEDURE TERM(I7,I,IT:INTEGER;DLC,DLS,DRC,DRS,DBC,DBS:REAL);
  BEGIN
   IF  IT=0  THEN  ADDTHE(C7[I7],S7[I7],C[I],S[I],U,V)
           ELSE BEGIN U:=U*T; V:=V*T END;
   DL:=DL+DLC*U+DLS*V; DR:=DR+DRC*U+DRS*V; DB:=DB+DBC*U+DBS*V;
 END;

PROCEDURE PERTJUP;(* возмущения, вызванные Юпитером          *)
  BEGIN
    C[0]:=1.0;    S[0]:=0.0;    C[-1]:=COS(M5);    S[-1]:=-SIN(M5);
    ADDTHE(C[-1],S[-1],C[-1],S[-1],C[-2],S[-2]);
    TERM(-1,-1,0,    0.0,    0.0,    -0.1,      1.7,    -0.1,    0.0);
    TERM( 0,-1,0,    0.5,   -1.2,    18.9,      9.1,    -0.9,    0.1);
    TERM( 1,-1,0,  -21.2,   48.7,  -455.5,   -198.8,     0.0,    0.0);
    TERM( 1,-2,0,   -0.5,    1.2,   -10.9,     -4.8,     0.0,    0.0);
    TERM( 2,-1,0,   -1.3,    3.2,   -23.2,    -11.1,     0.3,    0.1);
    TERM( 2,-2,0,   -0.2,    0.2,     1.1,      1.5,     0.0,    0.0);
    TERM( 3,-1,0,    0.0,    0.2,    -1.8,      0.4,     0.0,    0.0);
  END;

PROCEDURE PERTSAT;(* кеплеровские члены и возмущения,          *)
                  (* вызванные Сатурном                        *)
  VAR I: INTEGER;
  BEGIN
    C[-1]:=COS(M6);      S[-1]:=-SIN(M6);
FOR I:=-1 DOWNTO -3 DO ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
    TERM( 0,-1,0,    1.4,   -0.5,    -6.4,      9.0,    -0.4,   -0.8);
    TERM( 1,-1,0,  -18.6,  -12.6,    36.7,   -336.8,     1.0,    0.3);
    TERM( 1,-2,0,   -0.7,   -0.3,     0.5,     -7.5,     0.1,    0.0);
    TERM( 2,-1,0,   20.0, -141.6,  -587.1,   -107.0,     3.1,   -0.8);
    TERM( 2,-1,1,    1.0,    1 4,     5.8,     -4.0,     0.0,    0.0);
    TERM( 2,-2,0,    1.6,   -3.8,   -35.6,    -16.0,     0.0,    0.0);
    TERM( 3,-1,0,   75.3, -100.9,   128.9,     77.5,    -0.8,    0.1);
    TERM( 3,-1,1,    0.2,    1.8,    -1.9,      0.3,     0.0,    0.0);
```

```
    TERM( 3,-2,0,    2.3,   -1.3,   -9.5,    -17.9,   0.0,   0.1);
    TERM( 3,-3,0,   -0.7,   -0.5,   -4.9,      6.8,   0.0,   0.0);
    TERM( 4,-1,0,    3.4,   -5.0,   21.6,     14.3,  -0.8,  -0.5);
    TERM( 4,-2,0,    1.9,    0.1,    1.2,    -12.1,   0.0,   0.0);
    TERM( 4,-3,0,   -0.1,   -0.4,   -3.9,      1.2,   0.0,   0.0);
    TERM( 4,-4,0,   -0.2,    0.1,    1.6,      1.8,   0.0,   0.0);
    TERM( 5,-1,0,    0.2,   -0.3,    1.0,      0.6,  -0.1,   0.0);
    TERM( 5,-2,0,   -2.2,   -2.2,   -7.7,      8.5,   0.0,   0.0);
    TERM( 5,-3,0,    0.1,   -0.2,   -1.4,     -0.4,   0.0,   0.0);
    TERM( 5,-4,0,   -0.1,    0.0,    0.1,      1.2,   0.0,   0.0);
    TERM( 6,-2,0,   -0.2,   -0.6,    1.4,     -0.7,   0.0,   0.0);
  END;

PROCEDURE PERTNEP;(* возмущения, вызванные Нептуном          *)
  VAR I: INTEGER;
  BEGIN
    C[-1]:=COS(M8);      S[-1]:=-SIN(M8);
FOR  I:=-1  DOWNTO  -7  DO  ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
 TERM( 1,  0,0,  -78.1,  19518.1,  -90718.2,  -334.7,  2759.5,  -311.9);
 TERM( 1,  0,1,  -81.6,    107.7,    -497.4,  -379.5,    -2.8,   -43.7);
 TERM( 1,  0,2,   -6.6,     -3.1,      14.4,   -30.6,    -0.4,    -0.5);
 TERM( 1,  0,3,    0.0,     -0.5,       2.4,     0.0,     0.0,     0.0);
 TERM( 2,  0,0,   -2.4,    586.1,   -2145.2,   -15.3,   130.6,   -14.3);
 TERM( 2,  0,1,   -4.5,      6.6,     -24.2,   -17.8,     0.7,    -1.6);
 TERM( 2,  0,2,   -0.4,      0.0,       0.1,    -1.4,     0.0,     0.0);
 TERM( 3,  0,0,    0.0,     24.5,     -76.2,    -0.6,     7.0,    -0.7);
 TERM( 3,  0,1,   -0.2,      0.4,      -1.4,    -0.8,     0.1,    -0.1);
 TERM( 4,  0,0,    0.0,      1.1,      -3.0,     0.1,     0.4,     0.0);
 TERM(-1,-1,0,    -0.2,      0.2,       0.7,     0.7,    -0.1,     0.0);
 TERM( 0,-1,0,    -2.8,      2.5,       8.7,    10.5,    -0.4,    -0.1);
 TERM( 1,-1,0,   -28.4,     20.3,     -51.4,   -72.0,     0.0,     0.0);
 TERM( 1,-2,0,    -0.6,     -0.1,       4.2,   -14.6,     0.2,     0.4);
 TERM( 1,-3,0,     0.2,      0.5,       3.4,    -1.6,    -0.1,     0.1);
 TERM( 2,-1,0,    -1.8,      1.3,      -5.5,    -7.7,     0.0,     0.3);
 TERM( 2,-2,0,    29.4,     10.2,     -29.0,    83.2,     0.0,     0.0);
 TERM( 2,-3,0,     8.8,     17.8,     -41.9,    21.5,    -0.1,    -0.3);
 TERM( 2,-4,0,     0.0,      0.1,      -2.1,    -0.9,     0.1,     0.0);
 TERM( 3,-2,0,     1.5,      0.5,      -1.7,     5.1,     0.1,    -0.2);
 TERM( 3,-3,0,     4.4,     14.6,     -84.3,    25.2,     0.1,    -0.1);
 TERM( 3,-4,0,     2.4,     -4.5,      12.0,     6.2,     0.0,     0.0);
 TERM( 3,-5,0,     2.9,     -0.9,       2.1,     6.2,     0.0,     0.0);
 TERM( 4,-3,0,     0.3,      1.0,      -4.0,     1.1,     0.1,    -0.1);
 TERM( 4,-4,0,     2.1,     -2.7,      17.9,    14.0,     0.0,     0.0);
 TERM( 4,-5,0,     3.0,     -0.4,       2.3,    17.6,    -0.1,    -0.1);
 TERM( 4,-6,0,    -0.6,     -0.5,       1.1,    -1.6,     0.0,     0.0);
 TERM( 5,-4,0,     0.2,     -0.2,       1.0,     0.8,     0.0,     0.0);
 TERM( 5,-5,0,    -0.9,     -0.1,       0.6,    -7.1,     0.0,     0.0);
```

```
TERM( 5,-6,0,   -0.5,    -0.6,     3.8,    -3.6,    0.0,    0.0);
TERM( 5,-7,0,    0.0,    -0.5,     3.0,     0.1,    0.0,    0.0);
TERM( 6,-6,0,    0.2,     0.3,    -2.7,     1.6,    0.0,    0.0);
TERM( 6,-7,0,   -0.1,     0.2,    -2.0,    -0.4,    0.0,    0.0);
TERM( 7,-7,0,    0.1,    -0.2,     1.3,     0.5,    0.0,    0.0);
TERM( 7,-8,0,    0.1,     0.0,     0.4,     0.9,    0.0,    0.0);
   END;

PROCEDURE PERTJSU; (* возмущения, вызванные Юпитером и Сатурном*)
   VAR I: INTEGER;
   BEGIN
     C[-1]:=COS(M6);             S[-1]:=-SIN(M6);
     C[-4]:=COS(-4*M6+2*M5);     S[-4]:= SIN(-4*M6+2*M5);
FOR I:=-4 DOWNTO -5 DO ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
     TERM(-2,-4,0,  -0.7,    0.4,   -1.5,    -2.5,   0.0,   0.0);
     TERM(-1,-4,0,  -0.1,   -0.1,   -2.2,     1.0,   0.0,   0.0);
     TERM( 1,-5,0,   0.1,   -0.4,    1.4,     0.2,   0.0,   0.0);
     TERM( 1,-6,0,   0.4,    0.5,   -0.8,    -0.8,   0.0,   0.0);
     TERM( 2,-6,0,   5.7,    6.3,   28.5,   -25.5,   0.0,   0.0);
     TERM( 2,-6,1,   0.1,   -0.2,   -1.1,    -0.6,   0.0,   0.0);
     TERM( 3,-6,0,  -1.4,   29.2,  -11.4,     1.1,   0.0,   0.0);
     TERM( 3,-6,1,   0.8,   -0.4,    0.2,     0.3,   0.0,   0.0);
     TERM( 4,-6,0,   0.0,    1.3,   -6.0,    -0.1,   0.0,   0.0);
   END;

  BEGIN (* URA200         *)

   DL:=0.0;   DR:=0.0;   DB:=0.0;
   M5:=P2*FRAC(0.0564472+8.4302889*T);
   M6:=P2*FRAC(0.8829611+3.3947583*T);
   M7:=P2*FRAC(0.3967117+1.1902849*T);
   M8:=P2*FRAC(0.7216833+0.6068528*T);
   C7[0]:=1.0;   S7[0]:=0.0;   C7[1]:=COS(M7);   S7[1]:=SIN(M7);
  FOR I:=2 TO 7 DO ADDTHE(C7[I-1],S7[I-1],C7[1],S7[1],C7[I],S7[I]);
  FOR I:=1 TO 2 DO BEGIN C7[-I]:=C7[I]; S7[-I]:=-S7[I]; END;
   PERTJUP; PERTSAT; PERTNEP; PERTJSU;
   L:= 360.0*FRAC(0.4734843 + M7/P2
      +  ((5082.3+34 2*T)*T+DL)/1296.0E3   );
   R:= 19.211991 + (-0.000333-0.000005*T)*T   +   DR*1.0E-5;
   B:= (-130.61 + (-0.54+0.04*T)*T + DB ) / 3600.0;

  END;  (* URA200         *)
(*-----------------------------------------------------------*)
```

Нептун

```
(*-----------------------------------------------------------*)
(* NEP200: Нептун;эклиптические координаты L,B,R (град.и а.е.)  *)
```

```
(*            равноденствие даты                                          *)
(*           (T:  время в юлианских столетиях с J2000)                    *)
(*           (    = (JED-2451545.0)/36525          )                      *)
(*-------------------------------------------------------------------------*)
PROCEDURE NEP200(T:REAL;VAR L,B,R:REAL);

  CONST P2=6.283185307;

  VAR C8,S8          : ARRAY [ 0..6] OF REAL;
      C,S            : ARRAY [-6..0] OF REAL;
      M5,M6,M7,M8    : REAL;
      U,V, DL,DR,DB: REAL;
      I              : INTEGER;

  FUNCTION FRAC(X:REAL):REAL;
    BEGIN X:=X-TRUNC(X); IF (X<0) THEN X:=X+1.0; FRAC:=X  END;

  PROCEDURE ADDTHE(C1,S1,C2,S2:REAL; VAR C,S:REAL);
    BEGIN  C:=C1*C2-S1*S2;  S:=S1*C2+C1*S2;  END;

  PROCEDURE TERM(I1,I,IT:INTEGER;DLC,DLS,DRC,DRS,DBC,DBS:REAL);
    BEGIN
     IF   IT=0   THEN   ADDTHE(C8[I1],S8[I1],C[I],S[I],U,V)
                 ELSE BEGIN U:=U*T; V:=V*T END;
     DL:=DL+DLC*U+DLS*V; DR:=DR+DRC*U+DRS*V; DB:=DB+DBC*U+DBS*V;
    END;

  PROCEDURE PERTJUP; (* возмущения, вызванные Юпитером                    *)
    BEGIN
      C[0]:=1.0;    S[0]:=0.0;    C[-1]:=COS(M5);    S[-1]:=-SIN(M5);
      ADDTHE(C[-1],S[-1],C[-1],S[-1],C[-2],S[-2]);
      TERM( 0, -1,0,    0.1,    0.1,    -3.0,     1.8,   -0.3,   -0.3);
      TERM( 1,  0,0,    0.0,    0.0,   -15.9,     9.0,    0.0,    0.0);
      TERM( 1, -1,0,  -17.6,  -29.3,   416.1,  -250.0,    0.0,    0.0);
      TERM( 1, -2,0,   -0.4,   -0.7,    10.4,    -6.2,    0.0,    0.0);
      TERM( 2, -1,0,   -0.2,   -0.4,     2.4,    -1.4,    0.4,   -0.3);
    END;

  PROCEDURE PERTSAT;(* возмущения, вызванные Сатурном                     *)
    BEGIN
      C[0]:=1.0;    S[0]:=0.0;    C[-1]:=COS(M6);    S[-1]:=-SIN(M6);
      ADDTHE(C[-1],S[-1],C[-1],S[-1],C[-2],S[-2]);
      TERM( 0,-1,0,   -0.1,    0.0,     0.2,    -1.8,   -0.1,   -0.5);
      TERM( 1, 0,0,    0.0,    0.0,    -8.3,   -10.4,    0.0,    0.0);
      TERM( 1,-1,0,   13.6,  -12.7,   187.5,   201.1,    0.0,    0.0);
      TERM( 1,-2,0,    0.4,   -0.4,     4.5,     4.5,    0.0,    0.0);
      TERM( 2,-1,0,    0.4,   -0.1,     1.7,    -3.2,    0.2,    0.2);
```

```
      TERM( 2,-2,0,  -0.1,     0.0,     -0.2,      2.7,      0.0,     0.0);
    END;

  PROCEDURE PERTURA;(* кеплеровские члены и возмущения,           *)
                    (* вызванные Ураном                           *)
    VAR I: INTEGER;
    BEGIN
      C[0]:=1.0;   S[0]:=0.0;   C[-1]:=COS(M7);   S[-1]:=-SIN(M7);
FOR I:=-1 DOWNTO -5 DO ADDTHE(C[I],S[I],C[-1],S[-1],C[I-1],S[I-1]);
      TERM( 1, 0,0,  32.3,  3549.5,  -25880.2,  235.8,  -6360.5,  374.0);
      TERM( 1, 0,1,  31.2,    34.4,    -251.4,  227.4,     34.9,   29.3);
      TERM( 1, 0,2,  -1.4,     3.9,     -28.6,  -10.1,      0.0,   -0.9);
      TERM( 2, 0,0,   6.1,    68.0,    -111.4,    2.0,    -54.7,    3.7);
      TERM( 2, 0,1,   0.8,    -0.2,      -2.1,    2.0,     -0.2,    0.8);
      TERM( 3, 0,0,   0.1,     1.0,      -0.7,    0.0,     -0.8,    0.1);
      TERM( 0,-1,0,  -0.1,    -0.3,      -3.6,    0.0,      0.0,    0.0);
      TERM( 1, 0,0,   0.0,     0.0,       5.5,   -6.9,      0.1,    0.0);
      TERM( 1,-1,0,  -2.2,    -1.6,    -116.3,  163.6,      0.0,   -0.1);
      TERM( 1,-2,0,   0.2,     0.1,      -1.2,    0.4,      0.0,   -0.1);
      TERM( 2,-1,0,   4.2,    -1.1,      -4.4,  -34.6,     -0.2,    0.1);
      TERM( 2,-2,0,   8.6,    -2.9,     -33.4,  -97.0,      0.2,    0.1);
      TERM( 3,-1,0,   0.1,    -0.2,       2.1,   -1.2,      0.0,    0.1);
      TERM( 3,-2,0,  -4.6,     9.3,      38.2,   19.8,      0.1,    0.1);
      TERM( 3,-3,0,  -0.5,     1 7,      23.5,    7.0,      0.0,    0.0);
      TERM( 4,-2,0,   0.2,     0.8,       3.3,   -1.5,     -0.2,   -0.1);
      TERM( 4,-3,0,   0.9,     1.7,      17.9,   -9.1,     -0.1,    0.0);
      TERM( 4,-4,0,  -0.4,    -0.4,      -6.2,    4.8,      0.0,    0.0);
      TERM( 5,-3,0,  -1.6,    -0.5,      -2.2,    7.0,      0.0,    0.0);
      TERM( 5,-4,0,  -0.4,    -0.1,      -0.7,    5.5,      0.0,    0.0);
      TERM( 5,-5,0,   0.2,     0.0,       0.0,   -3.5,      0.0,    0.0);
      TERM( 6,-4,0,  -0.3,     0.2,       2.1,    2.7,      0.0,    0.0);
      TERM( 6,-5,0,   0.1,    -0.1,      -1.4,   -1.4,      0.0,    0.0);
      TERM( 6,-6,0,  -0.1,     0.1,       1.4,    0.7,      0.0,    0.0);
    END;

  BEGIN (* NEP200         *)

    DL  :=0.0;  DR:=0.0;  DB:=0.0;
    M5  :=P2*FRAC(0.0563867+8 4298907*T);
    M6  :=P2*FRAC(0.8825086+3.3957748*T);
    M7  :=P2*FRAC(0.3965358+1 1902851*T);
    M8  :=P2*FRAC(0.7214906+0.6068526*T);
    C8[0]:=1.0;   S8[0]:=0.0;   C8[1]:=COS(M8);   S8[1]:=SIN(M8);
  FOR I :=2TO  6  DO  ADDTHE(C8[I-1],S8[I-1],C8[1],S8[1],C8[I],S8[I]),
    PERTJUP; PERTSAT; PERTURA;
    L   := 360.0*FRAC(0.1254046 + M8/P2
          +  ((4982.8-21.3*T)*T+DL)/1296.0E3   );
```

```
  R   := 30.072984 + (0.001234+0.000003*T)     * T + DR*1.0E-5;
  B   := (    54.77 + ( 0.26 + 0.06*T) * T     +  DB ) / 3600.0;

 END;  (* NEP200 *)
(*-----------------------------------------------------------*)
```

Плутон

Построение подпрограммы PLU200 отличается от остальных подпрограмм. Координаты скачала вычисляются относительно неподвижной эклиптики 1950 г., а затем приводятся к равноденствию даты. Этот метод необходим из-за большого наклона орбиты. Полное разложение координат привело бы к возникновению многих вековых членов. Далее, нужно помнить, что PLU200 можно использовать для вычислений в интервале между 1890 и 2100 гг. Это обусловлено тем обстоятельством, что разложение в ряд делалось не исходя из теории возмущений, а с помощью фурье-анализа эфемерид, численно проинтегрированных в указанном промежутке времени. Ошибки вычисления координат до 1890 и после 2100 гг. резко возрастают и достигают значений более 0,5°.

```
(*-----------------------------------------------------------*)
(* PLU200: Плутон;эклиптические координаты L,B,R (град.и а.е.) *)
(*         равноденствие даты; только для интервала          *)
(*         1890—2100 гг.!!                                   *)
(*         (T: время в юлианских столетиях с J2000)          *)
(*         ( = (JED-2451545.0)/36525               )         *)
(*-----------------------------------------------------------*)
PROCEDURE PLU200(T:REAL;VAR L,B,R:REAL);

 CONST P2=6.283185307;

 VAR C9,S9    : ARRAY [ 0..6] OF REAL;
     C,S      : ARRAY [-3..2] OF REAL;
     M5,M6,M9: REAL;
     DL,DR,DB: REAL;
     I        : INTEGER;

 FUNCTION FRAC(X:REAL):REAL;
   BEGIN  X:=X-TRUNC(X); IF (X<0) THEN X:=X+1.0; FRAC:=X   END;

 PROCEDURE ADDTHE(C1,S1,C2,S2:REAL; VAR C,S:REAL);
   BEGIN  C:=C1*C2-S1*S2;  S:=S1*C2+C1*S2;  END;

 PROCEDURE TERM(I9,I:INTEGER;DLC,DLS,DRC,DRS,DBC,DBS:REAL);
   VAR U,V: REAL;
   BEGIN
    ADDTHE(C9[I9],S9[I9],C[I],S[I],U,V);
    DL:=DL+DLC*U+DLS*V; DR:=DR+DRC*U+DRS*V; DB:=DB+DBC*U+DBS*V;
```

```
    END;

   PROCEDURE PERTJUP;(* кеплеровские члены и возмущения,         *)
                     (* вызванные Юпитером                       *)
    VAR I: INTEGER;
  BEGIN
   C[0] :=1.0; S[0]:=0.0;        C[1]:=COS(M5);     S[1]:=SIN(M5);
   FOR   I:=  0  DOWNTO-1DO   ADDTHE(C[I],S[I],C[1],-S[1],C[I-1],S[I-1]);
     ADDTHE(C[1],S[1],C[1],S[1],C[2],S[2]);
  TERM(1, 0,    0.06,  100924.08, -960396.0, 15965.1, 51987.68, -24288.76);
  TERM(2, 0,3274.74,    17835.12, -118252.2,  3632.4, 12687.49,  -6049.72);
  TERM(3, 0,1543.52,     4631.99,  -21446.6,  1167.0,  3504.00,  -1853.10);
  TERM(4, 0, 688.99,     1227.08,   -4823.4,   213.5,  1048.19,   -648.26);
  TERM(5, 0, 242.27,      415.93,   -1075.4,   140.6,   302.33,   -209.76);
  TERM(6, 0, 138.41,      110.91,    -308.8,   -55.3,   109.52,    -93.82);
  TERM(3,-1,   -0.99,       5.06,     -25.6,    19.8,     1.26,     -1.96);
  TERM(2,-1,    7.15,       5.61,     -96.7,    57.2,     1.64,     -2.16);
  TERM(1,-1,   10.79,      23.13,    -390.4,   236.4,    -0.33,      0.86);
  TERM(0, 1,   -0.23,       4.43,     102.8,    63.2,     3.15,      0.34);
  TERM(1, 1,   -1.10,      -0.92,      11.8,    -2.3,     0.43,      0.14);
  TERM(2, 1,    0.62,       0.84,       2.3,     0.7,     0.05,     -0.04);
  TERM(3, 1,   -0.38,      -0.45,       1.2,    -0.8,     0.04,      0.05);
  TERM(4, 1,    0.17,       0.25,       0.0,     0.2,    -0.01,     -0.01);
  TERM(3,-2,    0.06,       0.07,      -0.6,     0.3,     0.03,     -0.03);
  TERM(2,-2,    0.13,       0.20,      -2.2,     1.5,     0.03,     -0.07);
  TERM(1,-2,    0.32,       0.49,      -9.4,     5.7,    -0.01,      0.03);
  TERM(0,-2,   -0.04,      -0.07,       2.6,    -1.5,     0.07,     -0.02);
  END;

   PROCEDURE PERTSAT;(* возмущения, вызванные Сатурном            *)
    VAR I: INTEGER;
  BEGIN
      C[1]:=COS(M6);     S[1]:=SIN(M6);
   FOR  I:=0  DOWNTO  -1  DO  ADDTHE(C[I],S[I],C[1],-S[1],C[I-1],S[I-1]);
   TERM(1,-1,  -29.47,   75.97,   -106.4,    -204.9,    -40.71,  -17.55);
   TERM(0, 1,  -13.88,   18.20,     42.6,     -46.1,      1.13,    0.43);
   TERM(1, 1,    5.81,  -23.48,     15.0,      -6.8,     -7.48,    3.07);
   TERM(2, 1,  -10.27,   14.16,     -7.9,       0.4,      2.43,   -0.09);
   TERM(3, 1,    6.86,  -10.66,      7.3,      -0.3,     -2.25,    0.69);
   TERM(2,-2,    4.32,    2.00,      0.0,      -2.2,     -0.24,    0.12);
   TERM(1,-2,   -5.04,   -0.83,     -9.2,      -3.1,      0.79,   -0.24);
   TERM(0,-2,    4.25,    2.48,     -5.9,      -3.3,      0.58,    0.02);
  END;

   PROCEDURE PERTJUS;(* возмущения, вызванные Юпитером и Сатурном *)
    VAR PHI,X,Y: REAL;
    BEGIN
```

```
      PHI:=(M5-M6);   X:=COS(PHI);   Y:=SIN(PHI);
      DL :=DL-9.11*X+0.12*Y;  DR:=DR-3.4*X-3.3*Y;
      DB :=DB+0.81*X+0.78*Y;
      ADDTHE(X,Y,C9[1],S9[1],X,Y);
      DL :=DL+5.92*X+0.25*Y;  DR:=DR+2.3*X-3.8*Y;
      DB :=DB-0.67*X-0.51*Y;
    END;

  PROCEDURE PREC(T:REAL;VAR L,B:REAL);(* прецессия           *)
                                      (* (1950->равн. даты)  *)
    CONST DEG=57.2957795;
    VAR   D,PPI,PI,P,C1,S1,C2,S2,C3,S3,X,Y,Z:   REAL;
    BEGIN
      D  :=T+0.5;     L  :=L/DEG; B:=B/DEG;
      PPI:=3.044;     PI :=2.28E-4*D;      P:=(0.0243764+5.39E-6*D)*D;
      C1 :=COS(PI);   C2:=COS(B);    C3:=COS(PPI-L);
      S1 :=SIN(PI);   S2:=SIN(B);    S3:=SIN(PPI-L);
      X  :=C2*C3;     Y :=C1*C2*S3-S1*S2;     Z:=S1*C2*S3+C1*S2;
      B  := DEG * ARCTAN( Z / SQRT((1.0-Z)*(1.0+Z)) );
      IF (X>0) THEN L:=360.0*FRAC((PPI+P-ARCTAN(Y/X))/P2)
               ELSE L:=360.0*FRAC((PPI+P-ARCTAN(Y/X))/P2+0.5);
    END;

  BEGIN (* PLU200          *)

    DL  :=0.0;   DR:=0.0;   DB:=0.0;
    M5  :=P2*FRAC(0.0565314+8.4302963*T);
    M6  :=P2*FRAC(0.8829867+3.3947688*T);
    M9  :=P2*FRAC(0.0385795+0.4026667*T);
    C9[0]:=1.0;   S9[0]:=0.0;   C9[1]:=COS(M9);   S9[1]:=SIN(M9);
 FOR  I:=2  TO  6  DO  ADDTHE(C9[I-1],S9[I-1],C9[1],S9[1],C9[I],S9[I]);
    PERTJUP; PERTSAT; PERTJUS;
    L:= 360.0*FRAC( 0.6232469 + M9/P2 + DL/1296.0E3 );
    R:= 40.7247248  +  DR * 1.0E-5;
    B:= -3.909434   +  DB / 3600.0;
    PREC(T,L,B);

  END;  (* PLU200          *)
(*-------------------------------------------------------------  *)
```

ОБОЗНАЧЕНИЯ

a	большая полуось
a	коэффициент места наблюдения по долготе
A	азимут
b	эклиптическая широта
b	коэффициент места наблюдения по широте
$\dot{b}$	изменение эклиптической широты во времени
B	эклиптическая широта Солнца
c	скорость света (c = 299792458 м/сек)
$c_i(E)$	функции Штумпфа
d	диаметр земной тени
db	возмущение эклиптической широты
dl	возмущение эклиптической долготы
dr	возмущение расстояния
D	средняя элонгация Луны от Солнца
D	диаметр полутени Земли
e	эксцентриситет
$\mathbf{e}$	вектор единичной длины
E	эксцентрическая аномалия
ET	эфемеридное время
f	сплюснутость Земли
f	угол раствора тени или полутени
f	разность координат на фундаментальной плоскости
F	сила притяжения двух тел
F	среднее расстояние Луны от восходящего узла ее орбиты
F	фокусное расстояние
g	разность координат на фундаментальной плоскости
G	постоянная тяготения ($G = 2{,}959122083 \cdot 10^{-4}$ а.е.$^3 M_\odot^{-1} d^{-2}$)
GMST	среднее гринвичское звездное время
h	высота над горизонтом
H	эксцентрическая аномалия гиперболической орбиты
i	наклон орбиты к эклиптике
JD	юлианская дата
k	постоянная тяготения Гаусса (k = 0,01720209895)
k	отношение лунного радиуса к земному ($k \approx 0.2725$)
l	эклиптическая долгота
l	средняя аномалия Луны
l	расстояние вершины конуса тени от центра Земли
l'	средняя аномалия Солнца
$\dot{l}$	изменение эклиптической долготы во времени

L	эклиптическая долгота Солнца
L_0	средняя долгота Луны
M	средняя аномалия
M_h	средняя аномалия (гипербола)
$M_\odot$	масса Солнца
MJD	модифицированная юлианская дата
n	отношение площадей треугольников
n	суточное движение
p	параметр орбиты
$\mathbf{P}$	вектор Гаусса (направлен к перигелию)
q	расстояние перигелия
$\mathbf{Q}$	вектор Гаусса (перпендикулярный направлению на перигелий)
r	расстояние, удаление
$\mathbf{r}$	вектор положения
$\dot{r}$	изменение расстояния
R	расстояние Земля—Солнце
R	радиус небесного тела
R	приведение к эклиптике
R	рефракция
$\mathbf{R}$	вектор Гаусса (перпендикулярный плоскости орбиты)
$\mathbf{R}$	вектор геоцентрического положения Солнца
S	площадь сектора
t	время
T	время в юлианских столетиях с J2000
T	время обращения
T_0	момент прохождения перигелия
$T_n(x)$	полином Чебышева n-го порядка
ΔT	разность эфемеридного и всемирного времен
TDB	барицентрическое динамическое время
TDT	земное динамическое время
u	аргумент широты
U	вспомогательная величина для параболических орбит
UT	всемирное время
UTC	координированное всемирное время
$\mathbf{v}$	вектор скорости
x, y, z	прямоугольные координаты
z	вспомогательный угол при вычислении прецессии
X, Y	идеальные координаты (астрометрия)
α	прямое восхождение
β	эклиптическая широта
$\Delta\beta$	возмущение эклиптической широты
δ	склонение
Δ	геоцентрическое расстояние
Δ	площадь треугольника
ε	наклон эклиптики к экватору

$\Delta\varepsilon$	нутация в эклиптической широте
ζ	вспомогательный угол для вычисления прецессии
η	отношение площадей сектора и треугольника
ϑ	сферическая координата
ϑ	вспомогательный угол для вычисления прецессии
ϑ	позиционный угол
Θ	местное звездное время
Θ_0	звездное время в Гринвиче
λ	эклиптическая долгота
λ	географическая широта (здесь отсчитывается к западу)
$\Delta\lambda$	возмущение эклиптической долготы
Λ	вспомогательный угол для вычисления прецессии
ν	истинная аномалия
π	3,1415926 .
π	экваториальный горизонтальный параллакс
π	вспомогательный угол для вычисления прецессии
Π	вспомогательный угол для вычисления прецессии
Π	горизонтальный параллакс
$\bar{\omega}$	долгота перигелия
ρ	геоцентрическое расстояние
τ	часовой угол
τ	промежуток времени
φ	сферическая координата
φ	географическая широта
φ'	геоцентрическая широта
$\Delta\psi$	нутация в эклиптической долготе
ω	аргумент перигелия
Ω	долгота восходящего узла
	точка весеннего равноденствия
$\oplus$	Земля
$\odot$	Солнце
d	сутки
h	час
m	минуты
s	секунды
$\circ$	градусы

СЛОВАРЬ ТЕРМИНОВ

Аберрация. Смещение наблюдаемого положения светила относительно геометрического положения из-за конечной скорости света (*звездная аберрация, время распространения света*).

Азимут. Одна из координат в *горизонтальной системе координат* Это угол между точкой юга и направлением на светило, увеличивается в сторону запада

Астрометрические координаты. Координаты для сравнения с каталожными положениями звезд Астрометрические координаты обычно относятся к среднему *равноденствию* четко определенной эпохи (J2000, B1950) Для планет и комет астрометрические координаты даются с учетом *времени распространения света*

Астрометрия. Измерение положений светил с помощью разделенных кругов или фотографических снимков

Астрономическая единица. Единица длины, применяемая для измерения расстояний в Солнечной системе Одна астрономическая единица (а.е) приблизительно равна среднему расстоянию между Землей и Солнцем и составляет около 149,6 млн км

Видимые координаты. Координаты светила, которые нужны, например, для наведения на него телескопа с разделенными кругами. Видимые координаты связаны с действительной ориентацией земной оси и поэтому содержат поправки за *прецессию* и *нутацию* Кроме того, учитывается *звездная аберрация*, а для тел Солнечной системы также *время распространения света.*

Восходящий узел. Точка орбиты, в которой небесное тело пересекает эклиптику с юга на север

Время распространения света. За время, которое нужно свету, чтобы достичь Земли (одну астрономическую единицу свет проходит за 499 с), планета немного перемещается по орбите. Поэтому ее наблюдаемое положение не соответствует истинному положению в момент наблюдения

Всемирное время (UT) Исчисление времени, связанное с вращением Земли, основа исчисления гражданского времени. Всемирное время можно определить по наблюдению прохождений светил с известным *прямым восхождением* через меридиан. Так как скорость вращения Земли испытывает неправильные колебания, всемирное время не представляет собой равномерную временную шкалу (*эфемеридное время, динамическое время*)

Высота. В сферической астрономии этот термин означает угол между светилом и *горизонтом*, измеряемый наблюдателем на поверхности Земли. Из-за *рефракции* истинная высота светила отличается от наблюдаемой, причем разность может достичь 35′.

Гелиоцентрические координаты. Координаты относительно центра Солнца.

Географические координаты. Две величины (географическая широта и долгота), однозначно определяющие положение точки на поверхности Земли. Плоскостью отсчета географических координат служит экватор, началом отсчета долготы по международному соглашению выбран Гринвичский *меридиан.*

Геоцентрические координаты. Координаты относительно центра Земли.

Горизонт. Воображаемая линия пересечения плоскости, касательной к *земной поверхности в месте наблюдения,* с *небесной сферой. Горизонт* является плоскостью отсчета *горизонтальной системы координат,* с координатами *азимут* и *высота.*

Горизонтальная система координат. Система координат (*азимут* и *высота*), основная плоскость которой—локальный горизонт наблюдателя. Горизонтальную систему используют, например, при вычислении восходов и заходов светил. Однако она непригодна для вычисления эфемерид и опубликования их в ежегодниках, так как азимут и высота светила зависят от места наблюдения и из-за вращения Земли постоянно меняются.

Динамическое время (TDB, TDT). Физическая шкала времени, реализуемая, например, с помощью атомных часов. Динамическое время, как и *эфемеридное время,* представлено равномерной временной шкалой, учитывающей, однако, взаимосвязь пространства и времени в теории относительности. В то время как барицентрическое динамическое время (TDB) представляет собой время в центре масс Солнечной системы, земное динамическое время (TDT)—это время, которое показывали бы часы в центре Земли. Время в обеих шкалах отличается друг от друга и от *эфемеридного времени* всего на несколько миллисекунд.

Задача двух тел. Задача вычисления движения двух тел под влиянием взаимного притяжения. Задача двух тел применяется для описания в упрощенном виде движения планет, комет и астероидов вокруг Солнца, при этом влияние других тел Солнечной системы не учитывается. Важность задачи двух тел для вычисления эфемерид состоит в ее относительно простом математическом решении.

Задача многих тел. Задача вычисления движения многих тел с учетом взаимного тяготения. В то время как задача двух тел может быть полностью решена, для задачи многих тел это в общем виде невозможно. Приходится использовать аналитическую аппроксимацию (разложение в ряд) и численные методы.

Звездная аберрация. Различие между направлением распространения света для покоящегося относительно Солнца и движущегося вместе с Землей наблюдателей.

Звездное время. Прямое восхождение звезды, находящейся в меридиане наблюдателя. Звездные сутки соответствуют по продолжительности одному обороту Земли вокруг оси и длятся примерно 23^h56^m всемирного времени.

Зенит. Воображаемая точка на небесной сфере, расположенная прямо над наблюдателем. Высота зенита равна 90°.

Кульминация. Момент достижения светилом наибольшей (или наименьшей) высоты над *горизонтом*. Кульминация происходит при прохождении светила через *меридиан*.

Меридиан. В астрономии так называют большой круг небесной сферы, который проходит через *зенит* наблюдателя и пересекает горизонт в точках севера и юга (*кульминация*). В географии понятие «меридиан» обозначает большой круг на поверхности Земли, в плоскости которого лежит земная ось. Все меридианы проходят через полюса Земли и пересекают экватор под прямым углом. Меридианы служат для определения географической долготы места на Земле.

Нутация. Колебание земной оси вокруг своего среднего положения, накладывающееся на прецессию. Период нутации составляет 18,6 лет и определяется периодом оборота восходящего узла орбиты Луны.

Параллакс. Изменение положения светила относительно далеких неподвижных звезд из-за изменения положения наблюдателя. Чем меньше расстояние, тем больше параллакс, поэтому он заметен в основном для Луны и (в меньшей мере) для комет и астероидов, близких к Земле (*топоцентрические координаты*).

Перигей. Ближайшая к Земле точка орбиты Луны или спутника.

Перигелий. Ближайшая к Солнцу точка орбиты планеты или кометы.

Прецессия. Вековые смещения эклиптики и небесного экватора. Из-за возмущений, вызванных Солнцем и Луной, земная ось не неподвижна в пространстве. Она перемещается с периодом 26 000 лет по боковой поверхности конуса, описанного вокруг среднего полюса эклиптики. Соответственно меняется и ориентация небесного экватора. Влияние планет на орбиту Земли дает дополнительный вклад в вековое смещение эклиптики. Перемещение экватора, эклиптики и *точки весеннего равноденствия* требует указания момента отсчета (*равноденствия*) *экваториальных* и *эклиптических координат* светила.

Проблема Кеплера. Другое название задачи двух тел.

Прямое восхождение. Одна из координат системы *экваториальных координат*. Прямое восхождение отсчитывается от *точки весеннего равноденствия* вдоль *экватора* и растет к востоку. Полученный угол измеряют обычно в единицах времени—часах, минутах и секундах.

Равноденствие. Момент отсчета при указании астрономических координат. Так как положение эклиптики и экватора, служащих плоскостями отсчета в экваториальной и эклиптической системах координат, меняется во времени из-за *прецессии*, всякое указание координат должно дополняться указанием момента времени. Только при этом условии координаты определяются однозначно. Чаще всего используют равноденствие даты, равноденствие *B1950* и равноденствие *J2000*.

Рефракция. Преломление лучей света в атмосфере Земли. Наблюдателю на Земле светило кажется находящимся на большей высоте над горизонтом, чем было бы без рефракции. Вблизи горизонта лучу света приходится проходить особенно большой путь через атмосферу. Поэтому рефракция сильнее всего заметна при восходах или заходах светил.

Склонение. Угол между светилом и экватором, измеряемый по направлению, перпендикулярному *экватору*. Вместе с *прямым восхождением* является координатой системы *экваториальных координат. (Равноденствие).*

Топоцентрические координаты. Координаты, отнесенные к месту наблюдения на поверхности Земли. Топоцентрические и *геоцентрические координаты* различаются на величину *параллакса.*

Точка весеннего равноденствия. Точка пересечения эклиптики и экватора, в которой Солнце пересекает экватор с юга на север. В настоящее время это происходит 21 марта каждого года. Точный момент этого события определяет собой начало весны. Точка весеннего равноденствия является началом отсчета эклиптической долготы (*эклиптические координаты*) и *прямого восхождения.*

Экватор. Воображаемый большой круг небесной сферы, перпендикулярный оси вращения Земли. Экватор разделяет северную и южную половины небесной сферы и одновременно является основной плоскостью для системы *экваториальных координат—прямого восхождения* и *склонения.*

Экваториальные координаты. Координаты относительно экватора (*прямое восхождение* и *склонение*).

Эклиптика. Воображаемый большой круг на пересечении плоскости орбиты Земли и небесной сферы. Если наблюдать с Земли, Солнце за год проходит всю эклиптику один раз. Эклиптика является плоскостью отсчета в эклиптической системе координат, где координаты—эклиптическая долгота и широта (*эклиптические координаты*).

Эклиптические координаты. Положение светила по отношению к *эклиптике* определяется долготой, широтой и соответствующим *равноденствием.* Эклиптическая долгота отсчитывается от точки весеннего равноденствия по эклиптике к востоку. Эклиптическая широта—угол между светилом и эклиптикой, отсчитываемый в направлении, перпендикулярном эклиптике.

Элонгация. Угол, под которым наблюдатель видит два светила.

Эфемеридное время (ET). Равномерная временная шкала, используемая для вычисления координат планет, комет и астероидов. Эфемеридное время было введено, чтобы не зависеть от неравномерных и непредсказуемых колебаний вращения Земли—основы исчисления всемирного времени. Разница между всемирным временем (UT) и эфемеридным временем (ET) по определению равнялась нулю в начале XX в., а в настоящее время составляет около 1 мин.

Эфемериды. Таблица, в которой приведены координаты планеты, кометы или астероида в определенном промежутке времени.

B1950. Начало Бесселева года 1950 (B1950 = JD 2433282.423 = январь $0{,}923^d$, 1950) в астрономии долгое время использовалось как стандартная эпоха, сейчас заменено эпохой *J2000.*

J2000. Стандартная эпоха, широко используемая в астрономии. Она приходится на полдень 1 января 2000 г. (1,5 января 2000 = JD 2451545.0). Буква J обозначает юлианскую стандартную эпоху. Следующие одна за другой стандартные юлианские эпохи отличаются обычно на полное юлианское столетие, составляющее 36525 дней (например, J1900 = 2415020.0). (*B1950*).

СПИСОК ЛИТЕРАТУРЫ

Далее представлен список основных и подробных работ в области сферической астрономии и небесной механики, а также некоторые книги по программированию и вычислительной математике. Они должны помочь читателю, заинтересованному в более глубоком изучении отдельных вопросов, затронутых в этой книге.

Справочная литература общего характера

[1] Astronomical Almanac; U.S. Government Printing Office, Her Majesty's Stationery Office; Washington, London.
Основной ежегодник в англоязычных странах. Издание 1982 г. содержит список численных значений и констант (звездное время, прецессия и нутация, массы и диаметры планет, элементы физических эфемерид и т.д.), которые тогда были рекомендованы Международным астрономическим союзом для астрономических вычислений.

[2] H. Bucerius, M. Schneider; Himmelsmechanik I–II; Bd. 143/144, Bibliographisches Institut; Mannheim (1966).
Современное представление небесной механики. Рассмотрены наряду с другими темами общая теория возмущений, теория орбиты Луны и теория определения орбит.

[3] L.E. Dogett, G.H. Kaplan, P.K. Seidelmann; Almanac for Computers for the Year 19xx; Nautical Almanac Office, United States Naval Observatory; Washington.
Это ежегодное издание содержит координаты Солнца, Луны и планет в виде полиномов Чебышева, которые позволяют просто и точно рассчитать любую эфемериду на малой вычислительной машине. Во вступительной главе дано много ссылок и простых формул для вычисления астрономических событий.

[4] Explanatory Supplement to the American Ephemeris and Nautical Almanac; U.S. Government Printing Office, Her Majesty's Stationery Office; Washington, London (1974).
Дополнительный том к ежегодному изданию «Astronomical Almanac». Документация к используемым данным и методам вычислений.

[5] R.M. Green; Spherical Astronomy; Cambridge University Press; Cambridge(1985).
Современное стандартное пособие по сферической астрономии.

[6] R. Herschel, F. Piper; Pascal und Pascal-Systeme; Oldenburg Verlag; München, 5. Aufl. (1985).
Учебник программирования на Паскале.

[7] D. McNally; Positional Astronomy; Muller Educational; London (1974).
Введение в сферическую астрономию.

[8] Ж.Меес, Астрономические формулы для калькуляторов. М., «Мир», (1988).

Собрание используемых на практике наиболее интересных и примечательных вычислительных методов для любителей астрономии. Подробные примеры, которые можно попробовать решить на калькуляторе.

[9] J. Meeus; Astronomical Tables of the Sun, Moon and Planets; Willmann-Bell; Richmond, Virginia (1983).

Обширный список интересных астрономических данных и событий, таких, как фазы Луны, затмения Луны и Солнца, конфигурация планет, покрытия планет Луной.

[10] I.I. Mueller; Spherical and practical astronomy; Frederick Ungar Publishing Co.; New York (1969).

Сферическая астрономия и ее применение в геодезии. Среди прочего, рассматриваются основы и практическое применение различных определений времени, а также предсказания и расчет условий затмений Солнца и покрытий звезд.

[11] W. H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling; Numerical Recipes; Cambridge University Press; Cambridge (1986).

Обширное собрание различных методов из всех областей вычислительной математики. Даны программы на языках Фортран и Паскаль, приложена дискета.

[12] A.E. Roy; Orbital Motion; Adam Hilger Ltd; Bristol, 2nd ed. (1982).

Введение в небесную механику.

[13] H.R. Schwarz; Numerische Mathematik; B.G. Teubner Verlag; Stuttgart, 2. Aufl. (1988).

Изложение важнейших алгоритмов вычислительной математики, рассчитанное на практическое применение.

[14] K. Stumpff; Himmelsmechanik I-III; VEB Deutscher Verlag der Wissenschaften; Berlin (1959, 1965, 1974).

Общее изложение небесной механики.

К гл.2. (Системы координат)

[15] J.H. Lieske, T. Lederle, W. Fricke, B. Morando; Expressions for the Precession Quantities Based upon the IAU (1976) System of Astronomical Constants; Astronomy and Astrophysics, vol. 58, pp. 1—16 (1977).

Описание различных вспомогательных углов, которые необходимы для расчета прецессии в эклиптических и экваториальных координатах. Численные значения основаны на ныне действующей системе астрономических постоянных, введенной МАС в 1976 г.

[16] G. Moyer, The Origin of the Julian Day System; Sky and Telescope, vol. 61, pp.311-313 (April 1982).

Статья по истории юлианских дат с формулами для взаимного пе-

ресчета между непрерывным исчислением суток и обычным календарем.

К гл.3. (Восходы и заходы)

[17] L.D. Schmadel, G. Zech; Empirical transformations from UT to ET for the period 1800—1988; Astronomische Nachrichten, vol. 309, pp. 219—221 (1988).

Разложение наблюдаемой разности всемирного и эфемеридного времен в полином 12-го порядка с областью применимости почти в 200 лет.

Алгоритмы для вычисления восходов и заходов и учета рефракции даны в Almanac for Computers [3].

К гл.4. (Орбиты комет)

[18] J.M.A. Danby, T.M. Burkhardt; The Solution of Kepler's Equation I-II; Celestial Mechanics, vol. 31, pp. 95—107, pp. 317—328 (1983).

Обсуждение различных методов итераций для численного решения уравнений Кеплера и соответствующих исходных значений.

[19] B. Marsden; Catalogue of Cometary Orbits; Smithonian Astrophysical Observatory; Cambridge, Mass.;

Постоянно перерабатываемые списки оскулирующих элементов орбит известных комет.

[20] Эфемериды малых планет; Институт теоретической астрономии; С.-Петербург.

Элементы орбит и эфемериды известных малых планет. Издается ежегодно.

Вычисление орбит, близких к параболическим, по Штумпфу описано в т. I работы [14].

К гл.5. (Орбиты планет)

[21] P. Bretagnon, J.-L. Simon; Tables for the motion of the Sun and the five bright planets from -4000 to +2800; Tables for the motion of Uranus and Neptun from +1600 to +2800; Willmann-Bell; Richmond, Virginia (1986).

Разложение в ряд и программы, с помощью которых можно вычислить движения внутренних планет в течение большого времени с достаточной точностью. Координаты внешних планет представлены полиномами на отрезках времени.

[22] T. C. van Flandern, K.F. Pulkkinen; Low precision formulae for planetary positions; Astrophysical Journal Supplement Series, vol. 41, p.391; (1979).

Разложение в ряд для вычисления гелиоцентрических и геоцентрических координат планет с малой точностью (около 1′).

[23] E. Goffin, J. Meeus, C. Steyart; An accurate representation of the motion of Pluto; Astronomy and Astrophysics, vol. 155,

p. 323; (1986).

Представление орбиты Плутона в виде тригонометрического ряда, основанного на численном интегрировании эфемерид.

[24] G.W. Hill; Tables of Jupiter, Tables of Saturn; Astronomical Papers of the American Ephemeris, vol. VII, part 1—2; Washington (1898).

Аналитическое представление в виде ряда движения Юпитера и Сатурна

[25] M.P. Jarnagin, jr.; Expansions in elliptic motion; Astronomical Papers of the American Ephemeris, vol. XVIII; Washington (1965).

Представление в виде рядов уравнения центра, расстояния и других координат в проблеме Кеплера без учета возмущений до высоких порядков эксцентриситета.

[26] S. Newcomb; Tables of the motion of the Earth, Tables of the heliocentric motion of Mercury, Tables of the heliocentric motion of Venus, Tables of the heliocentric motion of Mars; Astronomical Papers of the American Ephemeris, vol. VI, part 1—4; Washington (1898).

Аналитическое разложение в ряд координат внутренних планет.

[27] S. Newcomb; Tables of the heliocentric motion of Uranus, Tables of the heliocentric motion of Neptun; Astronomical Papers of the American Ephemeris, vol. VII, part 3-4; Washington (1898).

Представление координат Урана и Нептуна в виде ряда.

[28] X.X Newhall, E.M. Standish Jr., J.G. Williams; DE102: a numerically integrated ephemeris of the Moon and planets spanning fourty-four centuries; Astronomy and Astrophysics, vol. 125, pp 150-167 (1983),

Описание лучших современных эфемерид на интервал времени в несколько тысяч лет.

[29] F.E Ross; New Elements of Mars; Astronomical Papers of the American Ephemeris, vol. IX, part 2; Washington (1917).

Уточнение элементов орбиты Марса из таблицы Ньюкомба.

К гл.6. (Орбита Луны)

[30] E.W Brown; An introductory treatise on the Lunar Theory; Cambridge University Press (1896), Dover Publications (1960).

Для аналитического рассмотрения движения Луны представлены различные методы теории возмущений.

[31] M. Chapront-Touzé, J. Chapront; ELP 2000-85: a semi-analytical lunar ephemeris adequate for historical times; Astronomy and Astrophysics, vol. 190, p. 342; (1988).

Средние элементы орбиты и возмущающие члены для представления орбиты Луны на протяжении большого промежутка времени.

[32] M.C. Gutzwiller, D.S. Schmidt; The motion of the Moon as com-

puted by the method of Hill, Brown and Eckert; Astronomical Papers of the American Ephemeris, vol. XXIII, part 1; Washington (1986).

Аналитическое представление в виде ряда основной проблемы движения Луны (без учета планетных возмущений).

[33] Improved Lunar Ephemeris 1952–1959; Nautical Almanac Office; Washington, 1954.

Переработанная в 1954 г. версия теории Луны Брауна со всеми необходимыми возмущающими членами и средними аргументами

Во втором томе книги Буцериуса и Шнейдера Himmelsmechanik II [2] можно найти хорошее введение в теорию орбиты Луны и основных возмущений Аппроксимация функций полиномами Чебышева обсуждается в учебниках вычислительной математики.

К гл.7. (Затмения Солнца)

[34] T. Oppolzer; Canon der Finsternisse; Denkschriften der Math.-Naturw.Classe der Kaiserlichen Akademie der Wissenschaften Bd. 52, Wien (1883); Dover Publications, Inc., New York (1962).

Элементы затмений Солнца и Луны в промежутке 1207—2163 гг. Карты положений центральной линии при полных солнечных затмениях.

[35] H. Mucke, J. Meeus; Canon of Solar Eclipses, -2003 to 2526; Astronomisches Büro, Hasenwartgasse 32, Wien (1984).

Элементы 10774 затмений Солнца за примерно 4500 лет, вычисленные на основе теории орбиты Солнца Ньюкомба и теории Луны Брауна. Простые ориентировочные карты положений центральной линии.

В книге Мееса «Астрономические формулы для калькулятора» [8] представлены различные методы, особенно удобные для быстрого предсказания и описания возможных затмений. Готовые таблицы солнечных затмений всех видов даны в [9] Классические методы вычисления затмений Солнца с помощью элементов Бесселя описаны в «Explanatory Supplement» [4] и у Мюллера [10]

К гл.8. (Покрытия звезд)

[36] J. Robertson; Catalog of 3539 Zodiacal Stars for the Equinox 1950.0. Astronomical Papers of the American Ephemeris, vol. X, part 2; Washington (1917).

Опорный каталог для прогноза и расчета покрытий звезд. Содержит близкие к эклиптике звезды приблизительно до девятой величины, которые могут быть покрыты Луной. Прогноз покрытий звезд описан в различных учебниках сферической астрономии (например, [5]) и в Explanatory Supplement [4]. Более детально исследование орбиты Луны и вращения Земли описано в книге Spherical and practical astronomy [10].

К гл.9. (Определение орбит)

[37] C.F. Gauss; Theory of the Motion of the Heavenly Bodies Moving about the Sun in Conic Sections; Dover Publications; New York;
Английский перевод книги «Theoria motus corporum coelestium», в которой Гаусс описывает свой метод определения орбиты. Книга в основном представляет исторический интерес, но мало годится как учебник по определению орбит.

[38] H. Bucerius; Bahnbestimmung als Randwertproblem I–V; Astronomische Nachrichten, vol 278, 280, 281, 282, (1950–1955);
Пять статей, в которых среди прочего рассматривается пригодность упрощенных и полных методов Гаусса для определения орбиты. На большом числе примеров показаны проблемы, возникающие при определении орбиты.

Современное введение в теорию определения орбиты предлагается и в гл. I книги «Himmelsmechanik I» Буцериуса и Шнайдера [2].

К гл.10. (Астрометрия)

[39] SAO Star Catalog; Smithsonian Astrophysical Observatory; Cambridge Massachusetts (1966);
Четырехтомный звездный каталог, в котором приведены положения, собственные движения и величины 258997 звезд. Эпоха и равноденствие 1950.0. Точность положения звезды составляет примерно 1″.

[40] W.Dieckvoss; AGK3 Star Catalogue of Positions and Proper Motions North of -2.5° Declination; Hamburg-Bergedorf (1975).
Обширный каталог положений звезд северного неба для задач астрометрии. Составлен по фотографическому обзору неба обсерваторий Бонн и Гамбург-Бергедорф. Равноденствие 1950.0.

[41] W. Tirion, B. Rappaport, G Lovi; Uranometria 2000.0; Willmann-Bell; Richmond, Virginia (1987).
Атлас южного и северного неба с 473 картами в масштабе 1,85 см = 1° для равноденствия 2000. Содержит звезды до $9{,}5^m$ из Боннского обозрения, южного Боннского обозрения и Кордобского обозрения. Соответствующий каталог звезд готовится к изданию.

Вычисление идеальных координат описано во многих учебниках сферической астрономии [5]. Подробности о методе наименьших квадратов можно найти в книгах по вычислительной математике, например в [11] и [13].

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ

Оглавление

Уважаемый читатель!
Ваши замечания о содержании книги, ее оформлении, качестве перевода и другие просим присылать по адресу: 129820, Москва, ГСП, 1-й Рижский пер., д. 2, издательство «Мир»

Учебное издание

ОЛИВЕР МОНТЕНБРУК,
ТОМАС ПФЛЕГЕР
Астрономия с персональным компьютером

Зав. редакцией А.Н. Матвеев
Зам. зав. редакцией В.В. Герасимовский
Ведущий редактор М.Ф. Путов
Редактор Г.Г. Сорокина
Художник Ю.Л. Максимов
Художественный редактор А. Волков
Технический редактор Г.М. Носкова
Корректор Т.М. Подгорная
ИБ № 7867

Оригинал-макет подготовлен на персональном компьютере и распечатан на лазерном принтере в издательстве «Мир»

Подписано к печати 8.12.92. Формат 60 × 88/16.
Бумага типографская № 1. Гарнитура литературная.
Объем 8,75 бум.л. Усл. печ. л. 17,15.
Усл. кр.-отт. 17,40. Уч.-изд. л. 17,54.
Изд. № 2/8238. Тираж 1 100 экз. Зак. 130
С 026.

Издательство «Мир» Министерства печати и информации Российской Федерации: 129820, ГСП, Москва, И-110, 1-й Рижский пер., 2

Московская типография № 9 НПО «Всесоюзная книжная палата»:
109033, Москва, Волочаевская ул., 40.