ПРОГНОЗ ДВИЖЕНИЯ МАЛЫХ ТЕЛ.
Простейшие алгоритмы.

Научно-технический отчёт

Содержание

Список таблиц

1 Алгоритмы операций с матрицами и векторами

1.1 Вычисление матриц поворота

Для преобразований между различными системами отсчёта используют операцию поворота системы координат и операцию переноса начальной точки.

Поворот системы координат выполняется с помощью матрицы поворота. Положительным направлением называется поворот против часовой стрелки.

Дан

угол α, действительное число, единица измерений – радианы.

Вычислить

матрицы поворота на угол α вокруг осей OX, OY, OZ.

Алгоритм вычислений.

Матрицу поворота на угол α вокруг оси OX обозначим $R_x(\alpha)$:

$$R_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{pmatrix}.$$

Матрицу поворота на угол α вокруг оси OY обозначим $R_y(\alpha)$:

$$R_y(\alpha) = \begin{pmatrix} \cos\alpha & 0 & -\sin\alpha \\ 0 & 1 & 0 \\ \sin\alpha & 0 & \cos\alpha \end{pmatrix}.$$

Матрицу поворота на угол α вокруг оси OZ обозначим $R_z(\alpha)$:

$$R_z(\alpha) = \begin{pmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Программная реализация алгоритма.

Модуль **unforfun.c**

void forrotmatr

1.2 Умножение матрицы на вектор

Результатом умножения матрицы на вектор является вектор.

Даны

матрица

$$R = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix},$$

вектор

$$\vec{r} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}.$$

Вычислить

Произведение

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \cdot \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}.$$

Алгоритм вычислений.

$$R \cdot \vec{r} = \begin{pmatrix} a_{11}r_1 + a_{12}r_2 + a_{13}r_3 \\ a_{21}r_1 + a_{22}r_2 + a_{23}r_3 \\ a_{31}r_1 + a_{32}r_2 + a_{33}r_3 \end{pmatrix}.$$

Программная реализация алгоритма.

Модуль **unforfun.c**

void multmatrvec

1.3 Умножение матрицы на матрицу

Результатом произведения двух матриц является новая матрица.

Даны

матрица

$$R_a = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix},$$

матрица

$$R_b = \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix}.$$

Вычислить

произведение

$$\begin{pmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \cdot \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix}.$$

Алгоритм вычислений.

$$\begin{aligned} c_{11} &= a_{11}b_{11} + a_{12}b_{21} + a_{13}b_{31}, \\ c_{12} &= a_{11}b_{12} + a_{12}b_{22} + a_{13}b_{32}, \\ c_{13} &= a_{11}b_{13} + a_{12}b_{23} + a_{13}b_{33}, \\ c_{21} &= a_{21}b_{11} + a_{22}b_{21} + a_{23}b_{31}, \\ c_{22} &= a_{21}b_{12} + a_{22}b_{22} + a_{23}b_{32}, \\ c_{23} &= a_{21}b_{13} + a_{22}b_{23} + a_{23}b_{33}, \\ c_{31} &= a_{31}b_{11} + a_{32}b_{21} + a_{33}b_{31}, \\ c_{32} &= a_{31}b_{12} + a_{32}b_{22} + a_{33}b_{32}, \\ c_{33} &= a_{31}b_{13} + a_{32}b_{23} + a_{33}b_{33}. \end{aligned}$$

Программная реализация алгоритма.

Модуль **unforfun.c**

void tomultmatr

1.4 Вычисление транспонированной матрицы

Результатом транспонирования матрицы является новая матрица. Произведение матрицы поворота на транспонированную матрицу равно единичной матрице.

Дана

матрица

$$R_a = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}.$$

Вычислить

матрицу

$$R_b = \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix},$$

транспонированную по отношению к матрице R_a.

Алгоритм вычислений.

$$\begin{aligned} b_{11} = a_{11}, \;\; b_{12} = a_{21}, \;\; b_{13} = a_{31}, \\ b_{21} = a_{12}, \;\; b_{22} = a_{22}, \;\; b_{23} = a_{32}, \\ b_{31} = a_{13}, \;\; b_{32} = a_{23}, \;\; b_{33} = a_{33}. \end{aligned}$$

Программная реализация алгоритма.

Модуль **unforfun.c**

void transpmatr

1.5 Файл unforfun.h – прототипы функций

```
double  sqr ( double a ) ;

double  frac ( double a ) ;

double  datan2 ( double sa , double ca ) ;

double  atandegree ( double s , double c  ) ;

double  scalarmult ( double a[4] , double b[4] ) ;

void forrotmatr ( int iaxis , double angle , double matr[4][4] ) ;

void tomultmatr ( double ma[4][4], double mb[4][4], double mc[4][4] ) ;

void multmatrvec ( double p[4][4], double v[4], double vp[4] ) ;

void transpmatr ( double ma[4][4], double mb[4][4] ) ;
```

2 Алгоритмы преобразования даты

2.1 Календарная дата и модифицированный юлианский день

Одним из важных параметров алгоритмов является время. Время может быть выражено в самых разнообразных формах. В вычислениях наиболее удобная форма — сквозная нумерация суток. Счёт юлианских дней идёт от 1 января 4713 года до нашей эры. Юлианские сутки начинаются в полдень.

Модифицированные юлианские дни образованы простым вычитанием из значения юлианских дней числа 2400000.5. Модифицированные юлианские сутки начинаются в гринвичскую полночь.

Сквозная нумерация суток, необходимая для вычислений внутри всех алгоритмов, совершенно неприемлема для задания исходных данных и выдачи результатов расчётов. Для этих целей используют понятную и доступную всем форму календарных дат.

Задача *первого алгоритма* состоит в преобразовании от календарной даты к модифицированному юлианскому дню.

Момент всемирного координированного времени UTC соответствует моменту московского декретного времени, уменьшенному на три часа.

Дано:

текущая дата и момент времени в шкале всемирного координированного времени (UTC) в форме

год	—	целое число,
месяц	—	целое число,
день	—	целое число,
час	—	целое число,
минута	—	целое число,
секунда	—	действительное число.

Вычислить

соответствующую текущему моменту времени модифицированную юлианскую дату.

Алгоритм вычислений.

Дополнительные переменные:

$$year \quad \text{(целое число)}, \qquad month \quad \text{(целое число)}.$$

Результат вычислений обозначим mjd (действительное число).

Выполним операции присвоения

$$year = \text{год} - 1900,\ month = \text{месяц} - 3$$

Если $month < 0$

то $month = month + 12;\ year = year - 1\,.$

Числовое значение модифицированной юлианской даты получаем по формуле

$$\text{mjd} = 15078.0 + 365.0 \cdot year + \text{INT}\left(\frac{year}{4}\right) + \text{INT}\,(0.5 + 30.6 \cdot month);$$

$$\text{mjd} = \text{mjd} + \text{день} + \frac{\text{час}}{24} + \frac{\text{минута}}{1440} + \frac{\text{секунда}}{86400}.$$

где $\text{INT}\,(x)$ — целая часть числа $x\,.$

Задача *второго алгоритма* состоит в преобразовании модифицированного юлианского дня к календарной дате.

Момент всемирного координированного времени UTC соответствует моменту московского декретного времени, уменьшенному на три часа.

Дана

соответствующая текущему моменту времени в шкале UTC модифицированная юлианская дата mjd (действительное число).

Вычислить

календарную дату и момент времени в шкале всемирного координированного времени (UTC) в форме

год	—	целое число,
месяц	—	целое число,
день	—	целое число,
час	—	целое число,
минута	—	целое число,
секунда	—	действительное число.

Алгоритм вычислений.

Вспомогательные переменные

sp	—	действительное число,
rd	—	действительное число,
nd	—	целое число,
nz	—	целое число,
na	—	целое число,
nb	—	целое число,
ma	—	целое число.

Порядок вычислений:

$$
\begin{aligned}
rd &= \mathrm{INT}(\mathrm{mjd}) - 15078.0;\\
nd &= \mathrm{INT}(rd);\\
nz &= \mathrm{INT}\Big(\frac{rd}{1461.01}\Big);\\
na &= nd - 1461 \cdot nz;\\
nb &= \mathrm{INT}\Big(\frac{na}{365.25}\Big);\\
\text{год} &= 4 \cdot nz + nb + 1900.
\end{aligned}
$$

Если $na = 1461$,
то месяц $= 2$; день $= 29$;
иначе

$$
\begin{aligned}
nz &= na - 365 * nb;\\
ma &= \mathrm{INT}\Big(\frac{nz - 0.5}{30.6}\Big);\\
\text{месяц} &= ma + 3;\\
\text{день} &= nz - \mathrm{INT}(30.6 \cdot \text{месяц} - 91.3).
\end{aligned}
$$

Если месяц > 12,
то месяц $=$ месяц $- 12$; год $=$ год $+ 1$.
Далее

$$
\begin{aligned}
sp &= 24.0 \cdot (\mathrm{mjd} - \mathrm{INT}(\mathrm{mjd}));\\
\text{час} &= \mathrm{INT}(sp);\\
sp &= 60.0 \cdot (sp - \text{час});\\
\text{минута} &= \mathrm{INT}(sp);\\
\text{секунда} &= 60.0 \cdot (sp - \text{минута}).
\end{aligned}
$$

Алгоритмы дают правильный результат на интервале календарных дат со 2 марта 1900 года по 27 февраля 2100 года.

Программная реализация алгоритмов.

Модуль **unfordat.c**

double fromdatetomjd

void transmjdtodate

2.2 Файл unfordat.h – прототипы функций

```
double fromdatetomjd ( int nday , int nmonth , int nyear ,
                       int nhour , int nminute , double second ) ;

void fromsecparttohourminsec ( double secpart ,
                               int *inthour , int *intmin ,
                               double *second ) ;

void transmjdtodate ( double tmjd ,
                      int *nday , int *nmonth , int *nyear ,
                      int *nhour , int *nminute ,
                      double *second ) ;
```

2.3 Контрольный пример

```
 input
 moscow decret winter time
         29      day
          2      month
       2036      year
          5      hour
         45      minute
          0.000   second

 output
 moment in modified julian day
      64752.11458333  day with part in utc scale

 input
 moment in modified julian day
      64752.11458333  day with part in utc scale

 output
 moscow decret winter time
         29      day
          2      month
       2036      year
          5      hour
         45      minute
          0.000  second
```

3 Две шкалы равномерного времени

3.1 Вычисление земного времени

На поверхности Земли и в околоземном пространстве пользуются равномерной шкалой земного времени TT.

Дано:

текущая дата и время в шкале всемирного координированного времени (UTC) в форме

год	—	целое число,
месяц	—	целое число,
день	—	целое число,
час	—	целое число,
минута	—	целое число,
секунда	—	действительное число.

Вычислить

соответствующий момент земного времени TT в модифицированных юлианских днях.

Алгоритм вычислений.

С помощью алгоритма разд. 2.1 на с. 8 надо вычислить модифицированную юлианскую дату mjd, соответствующую заданной текущей дате и времени.

Таблица 1: Поправка к шкале всемирного времени

mjd	ΔT	дата
49534.0	61.184	1994 07 01
50083.0	62.184	1996 01 01
50630.0	63.184	1997 07 01
51178.0	64.184	1999 01 01
53736.0	65.184	2006 01 01

Далее на основе табл. 1 со с. 12 надо узнать поправку ΔT в секундах времени.

Момент в шкале земного времени равен

$$TT = \mathrm{mjd} + \frac{\Delta T}{86400}\,.$$

Программная реализация алгоритма.

Модуль **unfortim.c**

void transutctotdt

3.2 Вычисление барицентрического динамического времени

В пределах Солнечной системы пользуются равномерной шкалой барицентрического динамического времени TDB.

В этой шкале вычисляются положения Луны, Солнца и параметры прецессии и нутации.

Дан

Момент в шкале земного времени TT, выраженный в модифицированных юлианских днях.

Вычислить

Соответствующее значение барицентрического динамического времени TDB, выраженное в модифицированных юлианских днях.

Алгоритм вычислений.

Вспомогательные переменные:

d — интервал времени в юлианских столетиях, от стандартной эпохи J2000.0, начало которой соответствует 1.5 января 2000 года, до текущего момента,

g — приближённое значение аргумента перигелия Земли в радианах.

$$d = \frac{TT - 51544.5}{36525.0},$$
$$g = 0.017453 \cdot (357.258 + 35999.050 \cdot d),$$
$$TDB = TT + \frac{0.001658 \cdot \sin(g + 0.0167 \cdot \sin g)}{86400}.$$

Программная реализация алгоритма.

Модуль **unfortim.c**

void transtdttotdb

3.3 Файл unfortim.h – прототипы функций

```
void transutctotdt( double tinutc , double *tintdt , double *deltata ) ;

void transtdttotdb( double tintdt , double *tintdb ) ;
```

4 Прецессия и нутация

4.1 Вычисление параметров прецессии

Параметрами прецессии называются три угловые переменные: $\zeta_a(t)$, $\theta_a(t)$, $z_a(t)$. Аргументом t для вычисления параметров прецессии является барицентрическое динамическое время TDB, выраженное в модифицированных юлианских днях.

С помощью параметров прецессии вычисляют матрицу прецессии для преобразования от системы координат, соответствующей стандартной эпохе J2000.0, в систему подвижного экватора и мгновенной эклиптики, соответствующую заданной эпохе.

Дан

момент времени t в шкале барицентрического динамического времени TDB, выраженный в модифицированных юлианских днях.

Вычислить

параметры прецессии, то есть числовые значения трёх угловых переменных $\zeta_a(t)$, $\theta_a(t)$, $z_a(t)$.

Алгоритм вычислений.

В постоянной $r_s = 4.848136811095 \cdot 10^{-6}$ хранится число для перевода дуговых секунд в радианную меру.

Вспомогательная переменная t_c содержит время в юлианских столетиях, прошедшее от стандартной эпохи J2000.0, начало которой соответствует 1.5 января 2000 года.

$$t_c = \frac{t - 51544.5}{36525.0}$$

Числовые значения равны

$$\begin{aligned}
\zeta_a(t) &= r_s \cdot (2306.2181 + (0.30188 + 0.017998 \cdot t_c) \cdot t_c) \cdot t_c, \\
\theta_a(t) &= r_s \cdot (2004.3109 - (0.42665 + 0.041833 \cdot t_c) \cdot t_c) \cdot t_c, \\
z_a(t) &= r_s \cdot (2306.2181 + (1.09468 + 0.018203 \cdot t_c) \cdot t_c) \cdot t_c.
\end{aligned}$$

Единицы измерений.

Параметры прецессии выражены в радианах.

Программная реализация алгоритма.

Модуль **unforpnm.c**

void clcprecangles

4.2 Вычисление матрицы прецессии

Матрица прецессии необходима для преобразования от системы координат, соответствующей стандартной эпохе J2000.0, в систему среднего подвижного экватора и мгновенной эклиптики, соответствующую заданной эпохе.

Дан

момент времени t в шкале барицентрического динамического времени TDB.

Вычислить

матрицу прецессии $\mathbf{P}(t)$.

Алгоритм вычислений.

Матрица прецессии суть произведение трёх матриц поворота:

$$\mathbf{P}(t) = R_z(-z_a(t)) \cdot R_y(\theta_a(t)) \cdot R_z(-\zeta_a(t)).$$

На момент времени t вычисляются параметры прецессии $\zeta_a(t)$, $\theta_a(t)$, $z_a(t)$, выраженные в радианах (алгоритм разд. 4.1 на с. 14).

С помощью алгоритма (разд. 1.1 на с. 3) последовательно вычисляются три матрицы поворота

$$R_z(-\zeta_a(t)), \quad R_y(\theta_a(t)), \quad R_z(-z_a(t)).$$

Далее необходимо использовать алгоритм умножения матриц:

$$\mathbf{P}(t) = R_z(-z_a(t)) \cdot R, \qquad R = R_y(\theta_a(t)) \cdot R_z(-\zeta_a(t)),$$

где R — вспомогательная матрица.

Программная реализация алгоритма.

Модуль **unforpnm.c**

void clcprecangles

void clcprecmatr

4.3 Вычисление угла наклона эклиптики

Числовое значение угла наклона мгновенной эклиптики к среднему подвижному экватору $\varepsilon(t)$, заданное на момент барицентрического динамического времени t, необходимо для вычисления матрицы нутации.

Дан

момент времени t в шкале барицентрического динамического времени.

Вычислить

числовое значение $\varepsilon(t)$ угла наклона эклиптики к среднему экватору.

Алгоритм вычислений.

$$\varepsilon(t) = r_s \cdot (84381.448 - (46.815 + (0.0059 - 0.001813 \cdot t_c) \cdot t_c) \cdot t_c) .$$

Программая реализация алгоритма.

Модуль **unfornut.c**

double togetepsmean

4.4 Вычисление фундаментальных аргументов

Фундаментальными аргументами называются следующие угловые переменные:
λ — средняя долгота Луны,
l — средняя аномалия Луны,
l' — средняя аномалия Солнца,
F — средний аргумент широты Луны,
D — разность средних долгот Луны и Солнца,

Фундаментальные аргументы являются функциями барицентрического динамического времени t и используются для вычисления параметров нутации и в теориях движения Луны и Солнца.

Дан

момент времени t в шкале барицентрического динамического времени.

Вычислить

числовые значения фундаментальных аргументов

$$\lambda(t),\ \ l(t),\ \ l'(t),\ \ F(t),\ \ D(t).$$

Алгоритм вычислений.

$$\begin{aligned}
r_g &= 0.017453292519943296\,,\\
t_c &= \frac{t - 51544.5}{36525}\,,\\
\lambda(t) &= r_g \cdot (218.31643250\\
&\quad + (481267.8812772222 - (0.00161167 - 0.00000528 \cdot t_c) \cdot t_c) \cdot t_c)\,,\\
l(t) &= r_g \cdot (134.96298139\\
&\quad + (477198.8673980556 + (0.00869722 + 0.00001778 \cdot t_c) \cdot t_c) \cdot t_c)\,,\\
l'(t) &= r_g \cdot (357.52772333\\
&\quad + (35999.05034 - (0.00016028 + 0.00000333 \cdot t_c) \cdot t_c) \cdot t_c)\,,\\
F(t) &= r_g \cdot (93.27191028\\
&\quad + (483202.0175380555 - (0.00368250 - 0.00000306 \cdot t_c) \cdot t_c) \cdot t_c)\,,\\
D(t) &= r_g \cdot (297.85036306\\
&\quad + (445267.11148 - (0.00191417 - 0.00000528 \cdot t_c) \cdot t_c) \cdot t_c)\,.
\end{aligned}$$

Единицы измерений.

Фундаментальные аргументы выражены в радианах.

Программная реализация алгоритма.

Модуль **unfornut.c**

void clcfundarg

4.5 Вычисление параметров нутации

Параметрами нутации называются следующие угловые переменные:

$\Delta\psi$ — нутация в долготе,

$\Delta\varepsilon$ — нутация в наклоне.

Параметры нутации являются функциями барицентрического динамического времени и необходимы для вычисления матрицы нутации. Числовое значение нутации в долготе используется при вычислении истинного звёздного времени.

Дан

момент времени t в шкале барицентрического динамического времени.

Вычислить

параметры нутации $\Delta\psi(t)\,,\ \Delta\varepsilon(t)\,.$

Алгоритм вычислений.

На момент времени t с помощью алгоритма разд. 4.4 со с. 17 необходимо вычислить фундаментальные аргументы

$$\lambda(t),\ \ l(t),\ \ l'(t),\ \ F(t),\ \ D(t).$$

Далее надо использовать формулы:

$$\begin{aligned}\Delta\psi(t) = r_s \cdot \big[&(-17.1996 - 0.01742 \cdot t_c) \cdot \sin(\lambda - F)\\ &+ (0.2062 + 0.00002 \cdot t_c) \cdot \sin(2\lambda - 2F)\\ &+ 0.0046 \cdot \sin(\lambda - 2l + F)\\ &+ 0.0011 \cdot \sin(2l - 2F)\\ &- (1.3187 + 0.00016 \cdot t_c) \cdot \sin(2\lambda - 2D)\\ &+ (0.1426 - 0.00034 \cdot t_c) \cdot \sin l'\\ &- (0.0517 - 0.00012 \cdot t_c) \cdot \sin(2\lambda + l' - 2D)\\ &+ (0.0217 - 0.00005 \cdot t_c) \cdot \sin(2\lambda - l' - 2D)\\ &+ (0.0129 + 0.00001 \cdot t_c) \cdot \sin(\lambda + F - 2D)\\ &+ 0.0048 \cdot \sin(2l - 2D)\\ &- 0.0022 \cdot \sin(2F - 2D)\\ &+ \ldots\big]\,,\end{aligned}$$

$$\begin{aligned}\Delta\varepsilon(t) = r_s \cdot \big[&(9.2025 + 0.00089 \cdot t_c) \cdot \cos(\lambda - F) \\ &- (0.0895 - 0.00005 \cdot t_c) \cdot \cos(2\lambda - 2F) \\ &- 0.0024 \cdot \cos(\lambda - 2l + F) \\ &+ (0.5736 - 0.00031 \cdot t_c) \cdot \cos(2\lambda - 2D) \\ &+ (0.0054 - 0.00001 \cdot t_c) \cdot \cos l' \\ &+ (0.0224 - 0.00006 \cdot t_c) \cdot \cos(2\lambda + l' - 2D) \\ &- (0.0095 - 0.00003 \cdot t_c) \cdot \cos(2\lambda - l' - 2D) \\ &- 0.0070 \cdot \cos(\lambda + F - 2D) \\ &+ \ldots\big]\,,\end{aligned}$$

где

$$\begin{aligned} r_s &= 4.848136811095 \cdot 10^{-6}\,, \\ t_c &= \frac{t - 51544.5}{36525}\,. \end{aligned}$$

Единицы измерений.
Параметры нутации выражены в радианах.
Программная реализация алгоритма.
Модуль **unfornut.c**
void clcnut

4.6 Вычисление матрицы нутации

Матрица нутации необходима для преобразования от системы координат, соответствующей среднему подвижному экватору, в систему координат, соответствующей истинному экватору.

Дан

момент времени t в шкале барицентрического динамического времени.

Вычислить

матрицу нутации $\mathbf{N}(t)$.

Алгоритм вычислений.

Матрица нутации суть произведение трёх матриц поворота:

$$\mathbf{N}(t) = R_x(-\varepsilon - \Delta\varepsilon) \cdot R_z(-\Delta\psi) \cdot R_x(\varepsilon).$$

Вычислить угол наклона мгновенной эклиптики к среднему подвижному экватору $\varepsilon(t)$ (алгоритм разд. 4.3 на с. 16).

На момент времени t в шкале барицентрического динамического времени вычислить параметры нутации $\Delta\psi(t)$, $\Delta\varepsilon(t)$ (алгоритм разд. 4.5 на с. 18).

С помощью алгоритма (разд. 1.1 на с. 3) последовательно вычислить три матрицы поворота:

$$R_x(-\varepsilon - \Delta\varepsilon), \quad R_z(-\Delta\psi), \quad R_x(\varepsilon).$$

Далее необходимо использовать алгоритм умножения матриц (разд. 1.3 на с. 5):

$$R = R_z(-\Delta\psi(t)) \cdot R_x(\varepsilon(t)),$$
$$\mathbf{N}(t) = R_x(-\varepsilon(t) - \Delta\varepsilon(t)) \cdot R,$$

где R — вспомогательная матрица.

Программная реализация алгоритма.

Модуль **unforpnm.c**

void clcnutmatr

4.7 Прототипы функций

Файл **unforpnm.h** :

```
void clcprecangles ( double epoch , double tmjd ,
                     double *dzita0 , double *teta , double *zet ) ;

void clcprecmatr ( double epoch , double tmjd , double precmatr[4][4] ) ;

void clcnutmatr ( double deltapsi , double deltaeps , double epsmean ,
                  double matrnut[4][4] ) ;
```

Файл **unfornut.h** :

```
double togetepsmean ( double tmjd ) ;

void clcfundarg ( double tmjd , double fundarg[6] ) ;

void clcnut ( double tmjd , double  *deltapsi , double *deltaeps ) ;
```

4.8 Контрольный пример

```
input

59152.0  moment in modified julian day

output

precession matrix  P

 0.999987104372264  -0.004657817791984  -0.002023813872801
 0.004657817791642   0.999989152296765  -0.000004713477259
 0.002023813873587  -0.000004713139788   0.999997952075499

nutation   matrix  N

 0.999999995979347   0.000082275361072   0.000035666114213
-0.000082275089649   0.999999996586437  -0.000007611504096
-0.000035666740330   0.000007608569633   0.999999999334997

output

result of multiplication P*N

 0.999987555756883  -0.004575558874333  -0.001988112764180
 0.004575543743875   0.999989532071017  -0.000012158772523
 0.001988147585937   0.000003061924296   0.999998023627948
```

5 Звёздное время

5.1 Всемирное координированное время

Момент всемирного координированного времени UTC соответствует моменту московского декретного времени, уменьшенному на три часа.

Моменту всемирного координированного времени UTC соответствует модифицированная юлианская дата UTC_{mjd}.

Дано:

текущая дата и время в шкале всемирного координированного времени (UTC) в форме

год	—	целое число,
месяц	—	целое число,
день	—	целое число,
час	—	целое число,
минута	—	целое число,
секунда	—	действительное число.

Вычислить

соответствующий текущему моменту времени момент всемирного координированного времени в форме модифицированной юлианской даты UTC_{mjd}.

Алгоритм вычислений.

Для вычислений надо использовать алгоритм разд. 2.1 на с. 8.

Единицы измерений.

Величина UTC_{mjd} измеряется в юлианских днях.

5.2 Всемирное время

Всемирное время UT1 связано со всемирным координированным временем UTC формулой

$$\mathrm{UT1} = \mathrm{UTC} + \Delta\mathrm{UT}\,.$$

Разность

$$\Delta\mathrm{UT} = \mathrm{UT1} - \mathrm{UTC}$$

может быть определена только на основе наблюдений.

Единицы измерений.

Поправка всемирного времени $\Delta\mathrm{UT}$ измеряется в секундах.

5.3 Гринвичское среднее звёздное время

Гринвичское среднее звёздное время $S^m_\oplus$ является функцией всемирного времени UT1.

Разность ΔUT = UT1 - UTC всемирного времени и всемирного координированного времени может быть определена только на основе наблюдений.

Даны

момент всемирного координированного времени UTC_{mjd} и числовое значение поправки ΔUT, выраженное в секундах времени.

Вычислить

гринвичское среднее звёздное время.

Алгоритм вычислений.

Переменная t_c содержит текущее значение всемирного времени в модифицированных юлианских днях:

$$t_c = \mathrm{UTC}_{mjd} + \frac{\Delta\mathrm{UT}}{86400}\,.$$

Функция INT(x) вычисляет целую часть действительного числа x:

$$T_U = \frac{\mathrm{INT}(\,t_c) - 51544.5}{36525}\,,$$

где T_U – время, отсчитываемое в юлианских столетиях по 36525 суток в системе всемирного времени UT1 от эпохи 2000, январь 1, 12^h UT1 (MJD51544.5).

$$\begin{aligned} S^m_\oplus \text{ в } 0^h \text{ UT1 } &= 1.753368559233266 + (628.3319706888409 \\ &\quad + (6.770714\cdot 10^{-6} - 4.51\cdot 10^{-10}\cdot T_U)\cdot T_U)\cdot T_U\,, \end{aligned}$$

Промежуток звёздного времени от 0^h UT1 до момента, соответствующего моменту всемирного времени UT1, равен

$$S^m_\oplus = S^m_\oplus \text{ в } 0^h \text{ UT1 } + r\cdot\Big[\,t_c - \mathrm{INT}(\,t_c)\Big],$$

где

$$r = 6.300388098984891 + (3.707456\cdot 10^{-10} - 3.707\cdot 10^{-14}\cdot T_U)\cdot T_U.$$

Единицы измерений.

Гринвичское среднее звёздное время $S^m_\oplus$ измеряется в радианах.

Программная реализация алгоритма.

Модуль **unforsit.c**

double togetgmstime

5.4 Гринвичское истинное звёздное время

Гринвичское среднее звёздное время $S^m_\oplus$ является функцией всемирного времени UT1.

Разность ΔUT = UT1 - UTC всемирного времени и всемирного координированного времени может быть определена только на основе наблюдений.

Гринвичское истинное звёздное время $S_\oplus$ есть угол от истинной точки весеннего равноденствия до гринвичского меридиана, отсчитываемый вдоль истинного экватора.

Даны

момент времени UTC и числовое значение поправки ΔUT.

Вычислить

гринвичское истинное звёздное время $S_\oplus$.

Алгоритм вычислений.

На основе алгоритма разд. 5.3 на с. 24 с помощью исходных данных UTC, ΔUT определить числовое значение гринвичского среднего звёздного времени $S^m_\oplus$.

Далее на момент времени $t = \mathrm{UTC}_{mjd}$ надо вычислить угол наклона мгновенной эклиптики к среднему подвижному экватору $\varepsilon(t)$ (алгоритм разд. 4.3 на с. 16) и параметр нутации $\Delta\psi(t)$ (алгоритм разд. 4.5 на с. 18).

Формула для вычисления гринвичского истинного звёздного времени гласит

$$S_\oplus \;=\; S^m_\oplus + \Delta\psi\,\cos\varepsilon.$$

Единицы измерений.

Гринвичское истинное звёздное время $S_\oplus$ измеряется в радианах.

Программная реализация алгоритма.

Модуль **unforsit.c**

double togetgastime

5.5 Матрица вращения Земли

Матрица вращения Земли $\mathbf{R}$ является матрицей поворота вокруг оси OZ на угол, равный значению истинного звёздного времени $S_\oplus$.

Дано

гринвичское истинное звёздное время $S_\oplus$.

Вычислить

матрицу вращения $\mathbf{R}$.

Алгоритм вычислений.

С помощью алгоритма разд. 1.1 на с. 3 надо вычислить матрицу

$$\mathbf{R} = R_z(S_\oplus).$$

Программная реализация алгоритмов.

Модуль **unforsit.c**

double togetgmstime

double togetgastime

void earthrotmatr

5.6 Файл unforsit.h – прототипы функций

```
double  togetgmstime ( double tinutc , double deltaut ) ;

double  togetgastime ( double tinutc , double deltaut ) ;

void earthrotmatr ( double utc , double dut ,  double rz[4][4] ) ;
```

5.7 Контрольный пример

```
the Earth rotation matrix

input

moscow decret winter time
         7      day
        12      month
      1999      year
         5      hour
        45      minute
         0.000  second

universal time correction ut1-utc
         0.384  second of time

output

moment in modified julian day
     51519.11458333  day with part in utc scale

moment in terrestrial time scale
     51519.11532620  day with part in tt scale

          84381.481  obliquity in second of arc
            -14.939  d(psi)    in second of arc

greenwich mean sidereal time

     2.036644850536  in radian

greenwich true sidereal time

     2.036578399019  in radian

earth rotation matrix

    -0.449121697258     0.893470593278     0.000000000000
    -0.893470593278    -0.449121697258     0.000000000000
     0.000000000000     0.000000000000     1.000000000000
```

6 Алгоритмы вычисления матриц преобразований

6.1 От средней экваториальной системы координат к небесной

Дана

матрица прецессии $\mathbf{P}(t)$.

Вычислить

матрицу преобразования от средней подвижной системы координат к небесной системе координат $\mathbf{M}_p$.

Алгоритм вычислений.

Матрица $\mathbf{M}_p$ является транспонированной по отношению к матрице прецессии $\mathbf{P}(t)$.

Для вычисления надо воспользоваться алгоритмом транспонирования матрицы (разд. 1.4 на с. 6).

Программная реализация алгоритма.

Модуль **unforrom.c**

void frommeantofixequm

6.2 От небесной к истинной экваториальной системе координат

Матрица перехода между небесной системой координат и истинной экваториальной системой является произведением матрицы нутации $\mathbf{N}(t)$ на матрицу прецессии $\mathbf{P}(t)$.

Даны

матрицы прецессии $\mathbf{P}(t)$ и нутации $\mathbf{N}(t)$.

Вычислить

матрицу преобразования от небесной системы координат к истинной экваториальной системе координат $\mathbf{M}_{np}$.

Алгоритм вычислений.

Для вычисления надо воспользоваться алгоритмом разд. 1.3 на с. 5.

$$\mathbf{M}_{np} = \mathbf{N}(t) \cdot \mathbf{P}(t).$$

Программная реализация алгоритмов.

Модуль **unforrom.c**

void frommeantofixequm

void fromfixtotrueequm

6.3 От истинной экваториальной системы координат к небесной

Матрица перехода от истинной экваториальной системы координат к небесной системе координат является транспонированной по отношению к матрице перехода от небесной системы координат к истинной экваториальной.

Дана

матрица $\mathbf{M}_{np}$ (разд. 6.2 на с. 28).

Вычислить

матрицу преобразования от истинной экваториальной системы координат к небесной системе координат $\mathbf{M}_{pn}$.

Алгоритм вычислений.

Матрица $\mathbf{M}_{pn}$ является транспонированной по отношению к матрице $\mathbf{M}_{np}$.

Для вычисления надо воспользоваться алгоритмом транспонирования матрицы (разд. 1.4 на с. 6).

Программная реализация алгоритма.

Модуль **unforrom.c**

void fromtruetofixequm

6.4 Преобразование из небесной системы координат в земную

Матрицы перехода от небесной к земной системе координат вычисляется как произведение матрицы вращения Земли, матрицы нутации, матрицы прецессии.

Даны

матрица прецессии $\mathbf{P}(t)$ (разд. 4.2 на с. 15),

матрица нутации $\mathbf{N}(t)$ (разд. 4.6 на с. 20)

матрица вращения Земли $R_z(S_\oplus)$ (разд. 5.5 на с. 26)

Вычислить

матрицу перехода от небесной к земной системе координат $\mathbf{M}_{ct}$.

Алгоритм вычислений.

Необходимо воспользоваться алгоритмом умножения матриц (разд. 1.3 на с. 5)

$$R = \mathbf{N}(t) \cdot \mathbf{P}(t),$$
$$\mathbf{M}_{ct} = R_z(S_\oplus) \cdot R,$$

где R – вспомогательная матрица.

Программная реализация алгоритма.

Модуль **unforrom.c**

void fromtruetofixequm

void fromfixtoterram

6.5 Преобразование из земной системы координат в небесную

Дана

Матрица преобразования из небесной системы координат в земную $\mathbf{M}_{ct}$.

Вычислить

матрицу преобразования из земной системы координат в небесную $\mathbf{M}_{tc}$.

Алгоритм вычислений.

Матрица $\mathbf{M}_{tc}$ равна матрице, транспонированной по отношению к матрице $\mathbf{M}_{ct}$.

Для вычисления надо воспользоваться алгоритмом транспонирования матрицы (разд. 1.4 на с. 6).

Программная реализация алгоритмов.

Модуль **unforrom.c**

void frommeantofixequm

void fromfixtotrueequm

void fromtruetofixequm

void fromfixtoterram

void fromterratofixm

6.6 Файл unforrom.h – прототипы функций

```
void frommeantofixequm ( double tc , double pf[4][4] ) ;

void fromfixtotrueequm ( double tc , double pn[4][4] ) ;

void fromtruetofixequm ( double tc , double pn[4][4] ) ;

void fromfixtoterram ( double utc , double dut ,
                       double *tdb ,double ptm[4][4] ) ;

void fromterratofixm ( double ptm[4][4] , double pzm[4][4] ) ;
```

6.7 Контрольный пример

```
from terrestrial to celestial system

input

moment in modified julian day

     51519.11458333  day with part in utc scale

universal time correction ut1-utc

         0.384  second of time

moment in terrestrial time scale

     51519.11532620  day with part in tt scale

moment in ephemeris time scale

     51519.11532619  day with part in tdb scale

greenwich mean sidereal time

     2.036644850536  in radian

greenwich true sidereal time

     2.036578399019  in radian

from celestial to terrestrial system matrix

    -0.449194954530     0.893433765159     0.000009932203
    -0.893433764506    -0.449194953704    -0.000044796575
    -0.000035561277    -0.000028996161     0.999999998947

output

from terrestrial to celestial system matrix

    -0.449194954530    -0.893433764506    -0.000035561277
     0.893433765159    -0.449194953704    -0.000028996161
     0.000009932203    -0.000044796575     0.999999998947
```

7 Алгоритмы преобразования вектора состояния

7.1 Вектор состояния

Вектор состояния малой планеты включает в себя следующие компоненты:
начальную дату:

день — целое число,

месяц — целое число,

год — целое число;

набор оскулирующих кеплеровских элементов:

a — величину большой полуоси орбиты в астрономических единицах,

e — величину эксцентриситета орбиты,

i — величину угла наклонения орбиты в градусах,

Ω — величину долготы восходящего узла орбиты в градусах,

ω — величину аргумента перигея орбиты в градусах,

M_0 — величину средней аномалии орбиты в градусах.

Параметры i, Ω, ω даны в гелиоцентрической эклиптической системе координат. Числовое значение элемента M_0 относится к начальной дате.

При переходе от кеплеровских элементов к декартовым координатам и скоростям используют три угловые переменные: эксцентрическую аномалию E, истинную аномалию v и аргумент широты $u = v + \omega$. Средняя и эксцентрическая аномалии связаны между собой трансцедентным уравнением Кеплера

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

В формулах преобразования используется гелиоцентрическая гравитационная постоянная fM_S, измеряемая в $\mathrm{AU}^3/\mathrm{day}^2$, размерность координат – астрономическая единица, скоростей – астрономическая единица в сутки. Масса измеряется в единицах массы Солнца. Числовое значение величины fM_S равно квадрату гравитационной постоянной Гаусса

$$k = 0.01720209895.$$

Программная реализация алгоритмов.

Модуль **unforkep.c**,

double keplerequation — решение уравнения Кеплера с помощью итерационного метода Ньютона (разд. 7.2 на с. 33),

void fromkeplertocart — переход от кеплеровских элементов орбиты к декартовым векторам положения и скорости (разд. 7.2 на с. 33),

7.2 Формулы преобразования вектора состояния

Пусть на начальный момент времени **задан** начальный вектор состояния небесного тела в гелиоцентрической эклиптической системе отсчёта.

Требуется

определить прямоугольные координаты и скорости объекта на любой момент времени в гелиоцентрической экваториальной системе.

Вычислим момент времени t_a в модифицированных юлианских днях, соответствующий начальному моменту времени. Произвольный момент времени в модифицированных юлианских днях обозначим t_b.

По формуле

$$n = 57.2957795130823209 \cdot \frac{k}{\sqrt{a^3}},$$

$k = 0.01720209895$ – гравитационная постоянная Гаусса, определим числовое значение среднего движения в градусах за сутки.

Вычислим значение средней аномалии M в градусах на момент t_b:

$$M = M_0 + n \cdot (t_b - t_a).$$

Для решения уравнения Кеплера, то есть вычисления значения эксцентрической аномалии E, соответствующей вычисленному значению средней аномалии M, используют итерационный метод Ньютона.

Начальное значение выбирают по формуле

$$E_0 = M - e \sin M,$$

где числовое значение средней аномалии выражено в радианах. Формула для первой итерации имеет вид:

$$E_1 = E_0 - \frac{E_0 - e \sin E_0 - M}{1 - e \cos E_0}.$$

Формула для итерации с номером $n+1$ имеет вид:

$$E_{n+1} = E_n - \frac{E_n - e \sin E_n - M}{1 - e \cos E_n}.$$

Достаточно четырёх – пяти итераций:

$$E = E_{n+1}.$$

Численное значение эксцентрической аномалии задано в радианах.

Истинная аномалия v и аргумент широты $u = v + \omega$ определены формулами

$$\sin v = \frac{\sqrt{1-e^2}\sin E}{1 - e\cos E},$$
$$\cos v = \frac{\cos E - e}{1 - e\cos E},$$
$$\sin u = \sin v \cos\omega + \cos v \sin\omega,$$
$$\cos u = \cos v \cos\omega - \sin v \sin\omega.$$

Далее вычислим прямоугольные координаты $x,\ y,\ z$

$$x = r\left(\cos u\cos\Omega - \sin u\sin\Omega\cos i\right),$$
$$y = r\left(\cos u\sin\Omega + \sin u\cos\Omega\cos i\right),$$
$$z = r\ \sin u\sin i$$

и скорости $\dot{x},\ \dot{y},\ \dot{z}$

$$\dot{x} = \frac{x}{r}V_r + \left(-\sin u\cos\Omega - \cos u\sin\Omega\cos i\right)V_n,$$
$$\dot{y} = \frac{y}{r}V_r + \left(-\sin u\sin\Omega + \cos u\cos\Omega\cos i\right)V_n,$$
$$\dot{z} = \frac{z}{r}V_r + \cos u\sin i\,V_n,$$

где

$r = a\left(1 - e\cos E\right)$ – модуль расстояния (AU),
$p = a(1-e^2)$ – параметр орбиты (AU),

а радиальная и тангенциальная скорости равны

$$V_r = \frac{k}{\sqrt{p}}\,e\sin v,$$
$$V_n = \frac{k}{\sqrt{p}}\,(1 + e\cos v).$$

Вычисления по формулам эллиптической орбиты дадут вектор положения $\vec{r}_{ecl}$ и вектор скорости $\dot{\vec{r}}_{ecl}$ малой планеты относительно Солнца в эклиптической системе.

Вектор подожения $\vec{r}_{ecl}$ имеет координаты

$$x_{ecl},\ \ y_{ecl},\ \ z_{ecl}.$$

Вектор скорости $\dot{\vec{r}}_{ecl}$ имеет координаты

$$\dot{x}_{ecl},\ \ \dot{y}_{ecl},\ \ \dot{z}_{ecl}.$$

Вектор положения астероида $\vec{r}$ в гелиоцентрической экваториальной системе имеет координаты x, y, z, а вектор скорости определён составляющими $\dot{x}$, $\dot{y}$, $\dot{z}$.

Переход от эклиптической к экваториальной системе выполняется по формулам

$$\begin{aligned} x &= +x_{ecl}, \\ y &= +y_{ecl} \cdot \cos\varepsilon_A - z_{ecl} \cdot \sin\varepsilon_A, \\ z &= +y_{ecl} \cdot \sin\varepsilon_A + z_{ecl} \cdot \cos\varepsilon_A, \\ \dot{x} &= +\dot{x}_{ecl}, \\ \dot{y} &= +\dot{y}_{ecl} \cdot \cos\varepsilon_A - \dot{z}_{ecl} \cdot \sin\varepsilon_A, \\ \dot{z} &= +\dot{y}_{ecl} \cdot \sin\varepsilon_A + \dot{z}_{ecl} \cdot \cos\varepsilon_A. \end{aligned}$$

Числовое значение угла наклона эклиптики к экватору ε_A является одной из астрономических постоянных и вычисляется по формуле

$$\varepsilon_A = 23^\circ 26' 21''\!.448.$$

7.3 Формулы обратного преобразования

Пусть вектор положения и вектор скорости астероида в гелиоцентрической экваториальной системе имеют, соответственно, координаты

$$x_{equ},\ y_{equ},\ z_{equ},\ \dot{x}_{equ},\ \dot{y}_{equ},\ \dot{z}_{equ}.$$

В гелиоцентрической эклиптической системе положения x, y, z и скорости $\dot{x}$, $\dot{y}$, $\dot{z}$ объекта определены выражениями

$$\begin{aligned} x &= +x_{equ}, \\ y &= +y_{equ} \cdot \cos\varepsilon_A + z_{equ} \cdot \sin\varepsilon_A, \\ z &= -y_{equ} \cdot \sin\varepsilon_A + z_{equ} \cdot \cos\varepsilon_A, \\ \dot{x} &= +\dot{x}_{equ}, \\ \dot{y} &= +\dot{y}_{equ} \cdot \cos\varepsilon_A + \dot{z}_{equ} \cdot \sin\varepsilon_A, \\ \dot{z} &= -\dot{y}_{equ} \cdot \sin\varepsilon_A + \dot{z}_{equ} \cdot \cos\varepsilon_A. \end{aligned}$$

Для перехода от гелиоцентрических эклиптических координат и скоростей объекта

$$x,\ \ y,\ \ z,\ \ \dot{x},\ \ \dot{y},\ \ \dot{z}$$

к кеплеровским элементам орбиты

$$a,\ \ e,\ \ i,\ \ \Omega,\ \ \omega,\ \ M$$

служит следующий набор формул:

$r = \sqrt{x^2 + y^2 + z^2}$ – модуль расстояния (AU),

$V^2 = \dot{x}^2 + \dot{y}^2 + \dot{z}^2$ – квадрат модуля скорости $(\mathrm{AU/day})^2$,

$h = \dfrac{V^2}{2} - \dfrac{k^2}{r}$ – интеграл энергии $(\mathrm{AU/day})^2$,

$c_1 = y\dot{z} - z\dot{y}$ – первый интеграл площадей,

$c_2 = z\dot{x} - x\dot{z}$ – второй интеграл площадей,

$c_3 = x\dot{y} - y\dot{x}$ – третий интеграл площадей,

$l_1 = -\dfrac{k^2 x}{r} + \dot{y}c_3 - \dot{z}c_2$ – первый интеграл Лапласа,

$l_2 = -\dfrac{k^2 y}{r} + \dot{z}c_1 - \dot{x}c_3$ – второй интеграл Лапласа,

$l_3 = -\dfrac{k^2 z}{r} + \dot{x}c_2 - \dot{y}c_1$ – третий интеграл Лапласа,

$c = \sqrt{c_1^2 + c_2^2 + c_3^2}$ – модуль интеграла площадей $(\mathrm{AU}^2/\mathrm{day})$,

$l = \sqrt{l_1^2 + l_2^2 + l_3^2}$ – модуль интеграла Лапласа $(\mathrm{AU}^3/\mathrm{day}^2)$,

$a = -\dfrac{k^2}{2h}$ – большая полуось орбиты (AU),

$e = \dfrac{l}{k^2}$ – эксцентриситет орбиты,

$p = \dfrac{c^2}{k^2}$ – параметр орбиты (AU),

$\cos i = \dfrac{c_3}{c}$ – косинус угла наклонения орбиты,

$\sin i = \sqrt{1 - \cos^2 i}$ – синус угла наклонения орбиты,

$\sin\Omega = \dfrac{c_1}{c \sin i}$ – синус долготы восходящего узла,

$\cos\Omega = -\dfrac{c_2}{c \sin i}$ – косинус долготы восходящего узла,

$$\sin\omega = \frac{l_3}{l\sin i} \quad - \text{ синус аргумента перигея,}$$

$$\cos\omega = \frac{l_1}{l}\cos\Omega + \frac{l_2}{l}\sin\Omega \quad - \text{ косинус аргумента перигея,}$$

$$\sin u = \frac{z}{r\sin i} \quad - \text{ синус аргумента широты,}$$

$$\cos u = \frac{x}{r}\cos\Omega + \frac{y}{r}\sin\Omega \quad - \text{ косинус аргумента широты,}$$

$$\sin v = \sin u\cos\omega - \cos u\sin\omega \quad - \text{ синус истинной аномалии,}$$

$$\cos v = \cos u\cos\omega + \sin u\sin\omega \quad - \text{ косинус истинной аномалии,}$$

$$\sin E = \frac{\sqrt{1-e^2}\sin v}{1+e\cos v} \quad - \text{ синус эксцентрической аномалии,}$$

$$\cos E = \frac{\cos v + e}{1+e\cos v} \quad - \text{ косинус эксцентрической аномалии.}$$

Для вычисления угла наклонения орбиты i надо использовать функцию arctg от двух аргументов $\sin i$, $\cos i$ со значениями в области от 0 до 2π.

Для вычисления долготы восходящего узла орбиты Ω надо использовать функцию arctg от двух аргументов $\sin\Omega$, $\cos\Omega$ со значениями в области от 0 до 2π.

Для вычисления аргумента перигея орбиты ω надо использовать функцию arctg от двух аргументов $\sin\omega$, $\cos\omega$ со значениями в области от 0 до 2π.

Для вычисления истинной аномалии v надо использовать функцию arctg от двух аргументов $\sin v$, $\cos v$ со значениями в области от 0 до 2π.

Для определения значения средней аномалии орбиты M используют численное значение эксцентрической аномалии E:

$$E = v + \operatorname{arctg}\left(\frac{\sin E\cos v - \cos E\sin v}{\cos E\cos v + \sin E\sin v}\right),$$

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

Угловые переменные вектора состояния даны в градусах, в формулах использованы угловые переменные, выраженные в радианах.

Программная реализация алгоритмов.

Модуль **unforkep.c**,

void fromcarttokepler — переход от декартовых векторов положения и скорости к значениям кеплеровских элементов орбиты (разд. 7.3 на с. 35).

7.4 Файл unforkep.h – прототипы функций

```
/* for minor planet
   MeanMotion   : mean motion  n  in radian per day
   semi-major axis in AU
   Sqr(GaussConst)  in AU**3/(day**2*SolarMass)
   from the formula
   n^2*a^3=k^2
   or  n=k/sqrt(a^3) where  k = 0.01720209895 - Gauss const */

double fromaxistomot ( double a ) ;    /* semimajor axis in AU */

/* kepler equation   M=E-e*sin(E)
   to solve with the use Newton iteration method
   if  fi(x)=0  then iteration may be possible
   x_(n+1)=f(x_n)  where  f(x)=x-fi(x)/(d/dx(fi(x))) */

double keplerequation ( double eleme , double meananom ) ;

/* output   ecliptical system
   posvar 1,2,3 : a in AU, e,  inclination in degree
   angvar 1,2,3 : the ascending node in degree
                : the argument of perigei in degree
                : the mean anomaly in degree */

void fromcarttokepler ( double start ,                        /*  input */
                        double pos[4] , double vel[4] ,       /*  input */
                        double finish ,                       /*  input */
                        double posvar[4] , double angvar[4] ) ; /* output */

/* output   equatorial system
   position pos 1,2,3  : x,y,z in AU
   velosity vel 1,2,3  : vx,vy,vz in AU/day */

void fromkeplertocart ( double start ,                  /*  input */
                        double posvar[4] ,              /*  input */
                        double angvar[4] ,              /*  input */
                        double finish ,                 /*  input */
                        double pos[4] , double vel[4] ) ; /* output */
```

7.5 Контрольный пример

```
initial elements
ecliptic heliocentric

     53000.00000000  epoch in mjd
         1.91997795  a semimajor axis   AU
         0.43460482  e eccentricity
          11.878789  i inclination degree
         171.418697  O ascending node deg.
          26.433709  o arg.perihelium deg.
         335.308176  M mean anomaly degree

coordinate and velosity
equator heliocentric

     54000.00000000  moment in mjd
       -1.088981018  x  coordinate   in AU
        0.386338321  y  coordinate   in AU
        0.114106614  z  coordinate   in AU
    -0.002909948721  vx velosity in AU/day
    -0.018262604782  vy velosity in AU/day
    -0.003687289170  vz velosity in AU/day

the same elements
ecliptic heliocentric

     53000.00000000  epoch in mjd
         1.91997795  a semimajor axis   AU
         0.43460482  e eccentricity
          11.878789  i inclination degree
         171.418697  O ascending node deg.
          26.433709  o arg.perihelium deg.
         335.308176  M mean anomaly degree
```

8 Положения планет

8.1 Численные эфемериды

Для вычисления положений планет и Луны пользователям доступны численные эфемериды, полученные численным интегрированием уравнений движения планет Солнечной системы на больших интервалах времени. Информация о положении и скорости каждого объекта хранится в виде числовых значений коэффициентов полинома Чебышева.

Полиномы Чебышева $T_n(x)$ для значений аргумента

$$-1 \leqslant x \leqslant +1$$

вычисляются с помощью рекуррентного алгоритма

$$\begin{aligned} T_0(x) &= 1, \\ T_1(x) &= x, \\ T_n(x) &= 2\,x\,T_{n-1}(x) - T_{n-2}(x). \end{aligned}$$

Произвольная функция $f(t)$ на отрезке $t_1 \leqslant t \leqslant t_2$ при условии, что

$$x(t) = \frac{t - \dfrac{t_2 + t_1}{2}}{\dfrac{t_2 - t_1}{2}},$$

а максимальная степень полиномов равна m, может быть вычислена по формуле

$$f(t) = a_0 \cdot T_0(x(t)) + a_1 \cdot T_1(x(t)) + \ldots + a_m \cdot T_m(x(t))$$

где $a_0, a_1, \ldots a_m$ – численные коэффициенты.

Эфемеридные данные DE200/LE200 содержатся в файле прямого доступа в виде записей. Каждая запись содержит 745 чисел двойной точности в двоичном формате по 8 байт. В одной записи упакована информация о положениях и скоростях небесных объектов на интервале времени, равном 32 суткам. Первое число каждой записи содержит юлианскую дату начала очередного интервала прогнозирования.

Положения и скорости планет и Солнца даны относительно барицентра Солнечной системы.

Вместо координат Земли упакованы коэффициенты для вычисления положения центра масс системы Земля-Луна. Положение Луны дано относительно центра Земли.

Весь массив из 745 чисел расшифровывается с помощью пяти массивов целых чисел, представленных в табл.2.

В таблице

i – начальный номер в массиве,

j – конечный номер в массиве,

k – число коэффициентов аппроксимации,

l – интервал частной аппроксимации в днях внутри общего интервала,

m – количество аппроксимируемых переменных,

в последнем столбце даны отношения массы Солнца M_S к массе планеты M_N.

Таблица 2: Численные эфемериды

N	объект	i	j	k	l	m	M_S/M_N
1	Меркурий	2	145	12	8	3	6023600.0000
2	Венера	146	181	12	32	3	408523.5000
3	Земля-Луна	182	271	15	16	3	328900.5614
4	Марс	272	301	10	32	3	3098710.0000
5	Юпитер	302	338	9	32	3	1047.3500
6	Сатурн	339	352	8	32	3	3498.0000
7	Уран	353	476	8	32	3	22960.0000
8	Нептун	477	394	6	32	3	19314.0000
9	Плутон	395	412	6	32	3	130000000.0000
10	Луна	413	700	12	4	3	27068708.7489
11	Солнце	701	745	15	32	3	1.0000

Рассмотрим для примера алгоритм вычисления положения Земли относительно центра Солнца.

Пусть

t – дата в юлианских днях, выбранная для вычислений,

T_1 и T_2 – начальная и конечная даты эфемеридных данных.

Прежде всего определим порядковый номер того интервала, в который попадает выбранный нами момент $t_1 \leqslant t \leqslant t_2$:

$$n = \text{целому значению выражения } ((t - T_1)/32) + 1,$$

затем прочтём запись с этим номером из эфемеридного файла.

Для Солнца, центра масс системы Земля-Луна и Луны используем следующие

значения

$$\begin{aligned} i_{11} &= 701, \quad j_{11} = 745, \quad k_{11} = 15, \quad l_{11} = 32, \\ i_c &= 182, \quad j_c = 271, \quad k_c = 15, \quad l_c = 16, \\ i_{10} &= 413, \quad j_{10} = 700, \quad k_{10} = 12, \quad l_{10} = 4, \end{aligned}$$

с помощью которых выбираем из большого массива, содержащего 745 элементов, необходимые нам коэффициенты аппроксимации полиномами Чебышева.

Вычисляем положения Солнца $\vec{r}_{11}$ и центра масс Земля-Луна $\vec{r}_c$ относительно барицентра Солнечной системы и положение Луны $\vec{r}_{10}$ относительно центра Земли.

Вычисляем вектор положения Земли $\vec{r}_b$ относительно барицентра Солнечной системы

$$\vec{r}_b = \vec{r}_c - \frac{\mu}{1+\mu} \cdot \vec{r}_{10},$$

где коэффициент

$$\mu = 0.012300034$$

равен отношению массы Луны к массе Земли, а затем из этого вектора вычитаем соответствующий вектор для Солнца:

$$\vec{r}_3 = \vec{r}_b - \vec{r}_{11}.$$

Вектор положения Земли $\vec{r}_3$ относительно центра Солнца определён в экваториальной системе. Единицей измерения является километр. Числовые значения трёх координат вектора $\vec{r}_3$ необходимо разделить на числовое значение астрономической единицы.

Программная реализация алгоритмов.

Модуль **unephjpl.c**

int toopencoefile

void clcposvel

void clcephearthmoon

8.2 Файл unephjpl.h – прототипы функций

```
double  factormus[12] ;

double  factormuz[12] ;

double tiniteph,tlasteph ;

long  nofbyte,ibytemax ;

int  coefile, coebyte;

double  coefeph[745] ;

void toobtainfmus ( void ) ;

void clcposvel ( int  nplanet ,   /* input */
                 double  tintdb , /* input */
                 double  pos[4] ,
                 double  vel[4] ) ; /* output */

int toopencoefile ( void ) ; /* all variables are external */

void toreadephemeris ( double  tintdb ) ;

void clcephearthmoon ( double  tintdb , /* input */
                       double  posearth[4] ,
                       double  velearth[4] , /* output */
                       double  posmoon[4] ,
                       double  velmoon[4] ) ; /* output */

void earths ( double  tb ,             /*  input */
              double  x[4] ,
              double  v[4] ) ; /* output */
```

8.3 Контрольный пример

```
a try to open ephemeris file   0

the first record ephemeris file
       16.00       4208557.870944580994000            -0.000000049758453

the first moment and the last moment in ephemeris file
           16.00      104496.00

the moment to read ephemeris file     51550.00

 1       -0.0157146037773       -0.4124302728343       -0.2189183827777 AU
 1        0.0224927643812        0.0016303055756       -0.0014608292202 AU/day

 2       -0.7124583436473       -0.1497006396575       -0.0224414714450 AU
 2        0.0039069435935       -0.0180772977203       -0.0083796368453 AU/day

 3       -0.2779139100353        0.8646414950798        0.3750900188822 AU
 3       -0.0168203846191       -0.0044149810328       -0.0019135407775 AU/day

 4        1.3849915395478        0.0746548334154       -0.0031070951592 AU
 4       -0.0001616090105        0.0137836542532        0.0063267005983 AU/day

 5        3.9687941271309        2.7661484980882        1.0890153139375 AU
 5       -0.0046159401899        0.0058381325777        0.0026148831116 AU/day

 6        6.3756485427397        6.1913509059278        2.2828517150820 AU
 6       -0.0043004612336        0.0035085223832        0.0016340850254 AU/day

 7       14.4394987113804      -12.4954001100514       -5.6768979306772 AU
 7        0.0026805086789        0.0024578177110        0.0010385445137 AU/day

 8       16.8191802798003      -22.9735014098319       -9.8219137207224 AU
 8        0.0025836465122        0.0016630478881        0.0006163741824 AU/day

 9       -9.8664534969955      -27.9880670084820       -5.7615769897453 AU
 9        0.0030346526848       -0.0011327706700       -0.0012678310193 AU/day

10       -0.2770524516587        0.8622660573984        0.3741307547728 AU
10       -0.0162906217746       -0.0042223682427       -0.0018847973637 AU/day

11       -0.0071095289382       -0.0026793654791       -0.0009376115142 AU
11        0.0000054239976       -0.0000067173997       -0.0000030164210 AU/day
```

9 Интегрирование уравнений движения

9.1 Уравнения движения

Центр системы координат находится в центре Солнца. Ось абсцисс и ось ординат расположены в плоскости небесного экватора, фиксированного на эпоху J2000.0 (2000.0, январь, 1.5). Ось абсцисс направлена в точку весеннего равноденствия, соответствующую стандартной эпохе. Уравнения движения малой планеты записаны в этой системе координат, которую можно назвать гелиоцентрической экваториальной системой.

Начальные условия или начальный вектор состояния для каждой малой планеты известны в гелиоцентрической эклиптической системе координат. Положение эклиптики фиксировано на эпоху J2000.0 (2000.0, январь, 1.5). В этой системе в начальный момент времени задан вектор состояния малой планеты: набор оскулирующих кеплеровских элементов орбиты.

В алгоритмах используются следующие единицы измерений.

Единица расстояния – астрономическая единица (AU),

$$1 \text{ AU} = 149597870.691 \text{ км}.$$

Единица времени – эфемеридные сутки (day).

Единица массы – масса Солнца M_S.

Квадрат постоянной Гаусса $k = 0.01720209895$ равен гелиоцентрической гравитационной постоянной fM_S.

Гелицентрическая гравитационная постоянная измеряется в

$$\frac{(AU)^3}{(day)^2}.$$

Через $\vec{r}$ обозначим вектор положения малой планеты относительно центра Солнца в системе экватора и эклиптики, фиксированных на стандартную эпоху J2000.0. Уравнения движения имеют вид

$$\frac{d^2\vec{r}}{dt^2} = \vec{F}_S + \vec{F}_P,$$

где

$\vec{F}_S$ — ускорение, вызываемое Солнцем,

$\vec{F}_P$ — ускорение, обусловленное притяжением больших планет Солнечной системы и Луны.

Классическая часть ускорения вычисляются по формулам:

$$\vec{F}_S = -\frac{fM_S\,\vec{r}}{|\vec{r}|^3},$$

$$\vec{F}_P = \sum_{i=1}^{10} \vec{F}_i,$$

$$\vec{F}_i = -\frac{fm_i(\vec{r}-\vec{r}_i)}{|\vec{r}-\vec{r}_i|^3} - \frac{fm_i\vec{r}_i}{|\vec{r}_i|^3},$$

где

$\vec{r}_i$ – вектор положения большой планеты или Луны относительно Солнца,

m_i – масса возмущающего тела, $(i = 1, \ldots, 10)$.

Для преобразования начального вектора состояния малого тела в систему стандартного экватора надо использовать алгоритм 7.2 (с.33).

Числовые значения масс планет m_i в единицах массы Солнца можно вычислить на основе табл.2 (с.41).

9.2 Правые части уравнений

Алгоритм вычисления правых частей дифференциальных уравнений движения **реализован** в модуле

unequras.c

void planetsforces

void clcrighthandcoor

Прототипы функций:

```
void planetsforces ( double teph ,        /* input moment */
                     double curp[4] ,     /* input position */
                     double forcec[4] ) ; /* output forces */

void clcrighthandcoor ( double ut ,         /* input */
                        double rc[4] ,      /* input */
                        double force[4] ) ; /* output */
```

Контрольный пример

```
input
    53630.00  moment in modified julian day
position vector
        -0.664508850        -1.509827068        -0.815520967  AU
output
the force (acceleration) vector
   0.000031549664505   0.000071702302215   0.000038730021005  AU/day^2
```

9.3 Алгоритм интегрирования

Алгоритм интегрирования дифференциальных уравнений движения **реализован** в модуле

uneverim.c

void everinit

void evercoefnullo

void clcvectfrompolinom

void posvelevergrator

Прототипы функций:

```
void everinit (void) ;

void evercoefnullo (void) ;

void clcvectfrompolinom ( double dstep ,
                          double zeror[4] , double zerov[4] ,
                          double vectr[4] , double vectv[4] ) ;

void posvelevergrator ( double stepfull ,
                        double *tbeg , double zeror[4] , double zerov[4] ,
                        double *tend , double vectr[4] , double vectv[4] ) ;
```

Контрольный пример

```
to integrate orbit with minimum range from the Earth

   input

   minor body 1994 GK

   initial elements ecliptic heliocentric

        53000.00000000  epoch in mjd

            1.98661100  a semimajor axis   AU
            0.61091653  e eccentricity
              5.714540  i inclination degree
             15.205993  O ascending node deg.
            111.733859  o arg.perihelium deg.
            186.566521  M mean anomaly degree
```

```
 initial condition vector

 coordinate and velosity equator heliocentric

      53000.00000000  moment in mjd

         2.005833101  x  coordinate   in AU
        -2.152646860  y  coordinate   in AU
        -1.250913326  z  coordinate   in AU
      0.004465973888  vx velosity in AU/day
      0.003577829357  vy velosity in AU/day
      0.001846207706  vz velosity in AU/day

 output

 minor body 1994 GK

 r(AU) - distance from the Earth in astronomical unit

 r(km) - distance from the Earth in kilometer
```

date	h	a(AU)	e	i	r(AU)	r(km)
2008 3 27	12	1.987122	0.6110057	5.71538	0.04957	7415060.2
2008 3 28	0	1.987183	0.6110195	5.71554	0.04528	6774476.5
2008 3 28	12	1.987254	0.6110358	5.71574	0.04102	6136942.5
2008 3 29	0	1.987337	0.6110551	5.71600	0.03679	5503106.1
2008 3 29	12	1.987436	0.6110786	5.71635	0.03258	4873990.6
2008 3 30	0	1.987554	0.6111074	5.71681	0.02842	4251264.1
2008 3 30	12	1.987695	0.6111433	5.71748	0.02432	3637772.4
2008 3 31	0	1.987861	0.6111884	5.71850	0.02031	3038667.6
2008 3 31	12	1.988038	0.6112437	5.72014	0.01647	2464013.3
2008 4 1	0	1.988152	0.6113030	5.72306	0.01294	1935264.2
2008 4 1	12	1.987905	0.6113238	5.72853	0.01004	1501317.6
2008 4 2	0	1.986500	0.6111644	5.73775	0.00845	1263645.6
2008 4 2	12	1.983756	0.6107425	5.74712	0.00890	1331739.1
2008 4 3	0	1.981340	0.6103262	5.75197	0.01115	1668575.4
2008 4 3	12	1.979875	0.6100571	5.75370	0.01438	2151503.0
2008 4 4	0	1.979022	0.6098940	5.75425	0.01807	2703478.9
2008 4 4	12	1.978496	0.6097904	5.75439	0.02199	3290133.3
2008 4 5	0	1.978148	0.6097206	5.75438	0.02604	3896068.1
2008 4 5	12	1.977906	0.6096710	5.75432	0.03017	4513783.7
2008 4 6	0	1.977729	0.6096343	5.75425	0.03435	5139306.7
2008 4 6	12	1.977595	0.6096061	5.75417	0.03857	5770381.2
2008 4 7	0	1.977491	0.6095840	5.75410	0.04282	6405655.9
2008 4 7	12	1.977407	0.6095661	5.75403	0.04709	7044287.7