Семинар 5а. Решение практических задач

1. Решение задачи Коши для обыкновенного дифференциального уравнения (ОДУ).

$$\frac{\partial u}{\partial t} = f(u,t), \quad t > 0, \tag{1}$$

$$u(0) = u_0. \tag{2}$$

Приближенные методы решения в рамках метода сеток.

Основная идея метода сеток – поиск решения в конечном числе точек.

Интервал по времени $[0,T]$ заменяем множеством точек $\{t_n, n = 0,...,N_t\}$.

Чаще всего используются сетки равномерные, то есть с равноотстоящими узлами.

Тогда справедливо говорить о сетке с постоянным шагом τ – $\omega_t = \{t_n = \tau \cdot n, n = 0,...,N_t\}$.

Известные численные сеточные методы базируются на формуле Коши:

$$u(t) = u(0) + \int_0^t f(u(s),s)\,ds. \tag{3}$$

Если записать эту формулу для двух соседних точек сетки получим:

$$u(t_{n+1}) = u(t_n) + \int_{t_n}^{t_{n+1}} f(u(s),s)\,ds. \tag{4}$$

Далее для конструирования так называемой разностной схемы применяем разные квадратуры для аппроксимации интеграла в правой части:

1. Формула левых прямо-угольников	$u(t_{n+1}) \approx u(t_n) + f(u(t_n),t_n)\tau$. Компактная запись: $\frac{u^{n+1}-u^n}{\tau} = f^n, \quad f^n \equiv f(u^n,t_n)$.	Явная схема Эйлера 1-го порядка точности по τ
2. Формула правых прямо-угольников	$u(t_{n+1}) \approx u(t_n) + f(u(t_{n+1}),t_{n+1})\tau$. Компактная запись: $\frac{u^{n+1}-u^n}{\tau} = f^{n+1}, \quad f^{n+1} \equiv f(u^{n+1},t_{n+1})$.	Неявная схема Эйлера 1-го порядка точности по τ
3. Формула средних прямоу-гольников	$u(t_{n+1}) \approx u(t_n) + f\left(\left[u(t_n) + 0.5 f(u(t_n),t_n)\tau\right], t_n + 0.5\tau\right)\tau$. Компактная запись: $\frac{u^{n+1}-u^n}{\tau} = f^{n+1/2}, \quad f^{n+1/2} \equiv f(u^{n+1/2},t_{n+1/2}), \quad u^{n+1/2} = u^n + 0.5\tau f^n$.	Явная схема Рунге-Кутты 2-го порядка точности по τ
4. Формула трапеций	$u(t_{n+1}) \approx u(t_n) + 0.5\left[f(u(t_n),t_n) + f(u(t_{n+1}),t_{n+1})\right]\tau$. Компактная запись: $\frac{u^{n+1}-u^n}{\tau} = \frac{1}{2}\left(f^{n+1} + f^n\right)$.	Неявная схема Адамса 2-го порядка точности по τ

Дополнение к решению задачи по неявным схемам. В случае применения неявных схем может понадобиться итерационный процесс. Приведем примеры для неявной схемы Эйлера.

Пример 1. $f(u,t) = -u + \sin t$. В этом случае неявная схема Эйлера имеет вид:

$\frac{u^{n+1}-u^n}{\tau} = f^{n+1} = -u^{n+1} + \sin t_{n+1}$. Из нее следует: $u^{n+1} = \frac{u^n + \tau \sin t_{n+1}}{(1+\tau)}$, и итерации не требуются.

Пример 2. $f(u,t) = -\ln(1+u) + \sin t$. В этом случае неявная схема Эйлера имеет вид:

$\frac{u^{n+1}-u^n}{\tau} = f^{n+1} = -\ln(1+u^{n+1}) + \sin t_{n+1}$. Из нее следует: $u^{n+1} + \tau \ln u^{n+1} = u^n + \tau \sin t_{n+1}$, и итерации требуются.

Общая схема итераций для поиска решения по неявной схеме Эйлера:

$u^{n+1,0} = u^n; \quad \forall s = 0,1,2,...,m: \quad u^{n+1,s+1} = u^n + \tau f(u^{n+1,s}, t_{n+1})$.

Здесь $m = m(\varepsilon)$ – число итераций, необходимое для достижения точности ε.

Критерии останова итераций:

1) $|u^{n+1,s+1} - u^{n+1,s}| < \varepsilon$ – по абсолютной разности итераций;

2) $\left|u^{n+1,s+1}-u^{n+1,s}\right|<\varepsilon\left|u^{n+1,s}\right|$ – по относительной разности итераций;

3) $\left|R^{n+1,s}\right|<\varepsilon,\quad R^{n+1,s}=\dfrac{u^{n+1,s}-u^{n}}{\tau}-f\left(u^{n+1,s},t_{n+1}\right)$ – по невязке.

Пример 3. Реализация явной схемы Эйлера, вывод решения на экран.

Функция правой части $f(u,t)=u\sin(t)$. Параметр $u_0=1$. Точное решение $u(t)=\exp\left[1-\cos(t)\right]$.

Текст программы:

```
/*=============================================================*/
/* Программа "euler1.c"                                        */
/*=============================================================*/
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

// Заголовок функции правой части ОДУ:
double f(double u, double t);
double solution(double t);

// Головная программа:
int main() {
// Объявление переменных:
  int n, Nt=20;
  double T=1.0, u0 = 1.0;
  double tau = T/Nt;
  double t, u, err;
// Вычисления:
  for (n=0; n<=Nt; n++) {
    t = tau * n;
    if (n == 0) u = u0;
    else u = u + f(u,t)*tau;
    err = solution(t) - u;
    printf("t=%le u=%le err=%le\n", t, u, err);
  }
  return 0; // код возврата (ошибки)
}

// Реализация функции правой части ОДУ:
double f(double u, double t)
{
  return (u * sin(t));
}

// Реализация функции точного решения:
double solution(double t)
{
  return exp(1.0-cos(t));
}

Трансляция:
>gcc -o euler1 euler1.c -lm

Выполнение:
>euler1
t=0.000000e+00 u=1.000000e+00 err=0.000000e+00
t=5.000000e-02 u=1.002499e+00 err=-1.248438e-03
t=1.000000e-01 u=1.007503e+00 err=-2.494769e-03
t=1.500000e-01 u=1.015031e+00 err=-3.738869e-03
t=2.000000e-01 u=1.025114e+00 err=-4.980430e-03
t=2.500000e-01 u=1.037795e+00 err=-6.218867e-03
t=3.000000e-01 u=1.053129e+00 err=-7.453234e-03
```

```
t=3.500000e-01 u=1.071185e+00 err=-8.682129e-03
t=4.000000e-01 u=1.092042e+00 err=-9.903598e-03
t=4.500000e-01 u=1.115792e+00 err=-1.111504e-02
t=5.000000e-01 u=1.142539e+00 err=-1.231310e-02
t=5.500000e-01 u=1.172398e+00 err=-1.349356e-02
t=6.000000e-01 u=1.205498e+00 err=-1.465124e-02
t=6.500000e-01 u=1.241975e+00 err=-1.577987e-02
t=7.000000e-01 u=1.281980e+00 err=-1.687198e-02
t=7.500000e-01 u=1.325673e+00 err=-1.791880e-02
t=8.000000e-01 u=1.373222e+00 err=-1.891012e-02
t=8.500000e-01 u=1.424805e+00 err=-1.983419e-02
t=9.000000e-01 u=1.480610e+00 err=-2.067763e-02
t=9.500000e-01 u=1.540827e+00 err=-2.142533e-02
t=1.000000e+00 u=1.605656e+00 err=-2.206033e-02
```

Пример 4. Текст программы с выводом на экран и в текстовый файл:
Текст программы:

```
/*=============================================================*/
/* Программа "euler2.c"                                        */
/*=============================================================*/
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

// Заголовок функции правой части ОДУ:
double f(double u, double t);
double solution(double t);

// Головная программа:
int main() {
// Объявление переменных:
  int n, Nt=20;
  double T=1.0, u0 = 1.0;
  double tau = T/Nt;
  double t, u, err;
  FILE *FP;
// Откроем файл на запись:
  if ((FP = fopen("euler2.dat","w")) == NULL) {
    printf("File not open!");
    return -1;
  }
// Вычисления:
  for (n=0; n<=Nt; n++) {
    t = tau * n;
    if (n == 0) u = u0;
    else u = u + f(u,t)*tau;
    err = solution(t) - u;
    printf("t=%le u=%le err=%le\n", t, u, err);
    fprintf(FP, "%le %le\n", t, u);
  }
// Закроем файл:
  fclose(FP);
// Завершим программу:
  return 0; // код возврата
}

// Реализация функции правой части ОДУ:
double f(double u, double t)
{
  return (u * sin(t));
}
```

```
// Реализация функции точного решения:
double solution(double t)
{
  return exp(1.0-cos(t));
}

Трансляция:
>gcc -o euler2 euler2.c -lm
Выполнение:
>euler2
...
>cat euler2.dat
0.000000e+00 1.000000e+00
5.000000e-02 1.002499e+00
1.000000e-01 1.007503e+00
1.500000e-01 1.015031e+00
2.000000e-01 1.025114e+00
2.500000e-01 1.037795e+00
3.000000e-01 1.053129e+00
3.500000e-01 1.071185e+00
4.000000e-01 1.092042e+00
4.500000e-01 1.115792e+00
5.000000e-01 1.142539e+00
5.500000e-01 1.172398e+00
6.000000e-01 1.205498e+00
6.500000e-01 1.241975e+00
7.000000e-01 1.281980e+00
7.500000e-01 1.325673e+00
8.000000e-01 1.373222e+00
8.500000e-01 1.424805e+00
9.000000e-01 1.480610e+00
9.500000e-01 1.540827e+00
1.000000e+00 1.605656e+00
```

Пример 5. Реализация неявной схемы Эйлера, вывод результатов на экран и в файл.

```
/*================================================================*/
/* Программа "euler3.c"                                           */
/*================================================================*/
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
// Заголовок функции правой части ОДУ:
double f(double u, double t);
double solution(double t);
// Головная программа:
int main() {
// Объявление переменных:
  int n, Nt=60;
  double T=3.0, u0 = 1.0;
  double tau = T/Nt;
  double t, u, err;
  double err_min, err_max;
  FILE *FP;
// Откроем файл на запись:
  if ((FP = fopen("euler3.dat","w")) == NULL) {
    printf("File not open!");
    return -1; // Код возврата
  }
// Вычисления:
  for (n=0; n<=Nt; n++) {
    t = tau * n;
    if (n == 0) u = u0;
    else {
      // u^{n+1} = u^n + f(u^{n+1},t)*tau;
      // u^{n+1,0} = u^n; do: i=1,..,im: u^{n+1,i+1} = u^n + f(u^{n+1,i},t)*tau;
```

```
      int it=0, itm=100;
      double eps=1e-6;
      double un=u;
      do {
        it++;
        err = u;
        u = un + f(u,t)*tau;
        err = fabs(u - err);
        printf("it=%d abs_err=%le\n", it, err);
      } while ((err>eps) && (it<itm));
    }
    err = solution(t) - u;
    if (n>0) {
      double aerr = fabs(err);
      if (n==1) { err_min = aerr; err_max = aerr; }
      else      {
                  if (err_min > aerr) err_min = aerr;
                  if (err_max < aerr) err_max = aerr;
                }
    }
    printf("t=%le u=%le err=%le\n", t, u, err);
    fprintf(FP, "%le %le\n", t, u);
  }
// Закроем файл:
  fclose(FP);
// Завершим программу:
  printf("err_min=%le err_max=%le\n", err_min, err_max);
  return 0; // код возврата
}

// Реализация функции правой части ОДУ:
double f(double u, double t)
{
  return (u * sin(t));
}

// Реализация функции точного решения:
double solution(double t)
{
  return exp(1.0-cos(t));
}
```

Трансляция:

```
>gcc -o euler3 euler3.c -lm
```

Выполнение:

```
>euler3
t=0.000000e+00 u=1.000000e+00 err=0.000000e+00
it=1 abs_err=2.498958e-03
it=2 abs_err=6.244793e-06
it=3 abs_err=1.560548e-08
t=5.000000e-02 u=1.002505e+00 err=-1.254698e-03
it=1 abs_err=5.004176e-03
it=2 abs_err=2.497920e-05
it=3 abs_err=1.246879e-07
t=1.000000e-01 u=1.007534e+00 err=-2.526164e-03
it=1 abs_err=7.528204e-03
it=2 abs_err=5.625004e-05
it=3 abs_err=4.202950e-07
...
t=2.950000e+00 u=7.586961e+00 err=-3.318790e-01
it=1 abs_err=5.353360e-02
it=2 abs_err=3.777331e-04
it=3 abs_err=2.665285e-06
it=4 abs_err=1.880625e-08
t=3.000000e+00 u=7.640875e+00 err=-3.253963e-01
err_min=1.254698e-03 err_max=3.451643e-01
```

2. Решение начально-краевой задачи для одномерного уравнения переноса.

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0, \quad t > 0, \quad x > 0, \quad a = const > 0, \tag{1}$$

$$u(x,0) = u_0(x), \quad x \geq 0; \tag{2}$$

$$u(0,t) = \mu(t), \quad t > 0. \tag{3}$$

Условие согласования данных: $u_0(0) = \mu(0)$.

Явные численные схемы

Схема (А)	Схема (Б)
$\frac{u_i^{n+1} - u_i^n}{\tau} + a\frac{u_i^n - u_{i-1}^n}{h} = 0$	$\frac{u_i^{n+1} - \frac{1}{2}\left(u_{i+1}^n + u_{i-1}^n\right)}{\tau} + a\frac{u_{i+1}^n - u_{i-1}^n}{2h} = 0$
Расчетные формулы: $u_i^0 = u_0\left(x_i\right), \quad i = 0,...,N_x;$ $\forall n = 0,1,2,... : \quad u_0^{n+1} = \mu\left(t_{n+1}\right),$ $u_i^{n+1} = u_i^n - \gamma\left(u_i^n - u_{i-1}^n\right), \quad i > 0, \quad \gamma = \frac{\tau a}{h}.$	Расчетные формулы: $u_i^0 = u_0\left(x_i\right), \quad i = 0,...,N_x;$ $\forall n = 0,1,2,... : \quad u_0^{n+1} = \mu\left(t_{n+1}\right),$ $u_i^{n+1} = \frac{1}{2}\left(u_{i+1}^n + u_{i-1}^n\right) - \frac{1}{2}\gamma\left(u_{i+1}^n - u_{i-1}^n\right), \quad i > 0.$

Остановимся подробнее на схеме (А).

Входные данные программы:

u(x,0) = u0(x), u(0,t) = mu(t)

Например:

u0(x) = Sin(pi*x), 0<=x<=1, mu(t)=0

Входные параметры:

a=1, lx=1, Lx=10, t_max=1, Nx = 100

Производные параметры:

hx = Lx/Nx, tau = hx/a, Nt = t_max/tau, gam=tau*a/hx

Массивы: x(i), i=0,...,Nx; u(i), i=0,...,Nx.

Выходные данные:

Значения вектора u как функции вектора x в различные моменты времени.

Подготовка расчета:

задание нач. данных

заполнение нач. профиля

Цикл по времени:

пересчет профиля u(i) по разностной схеме

u_i = u_i - gam*(u_(i)-u_(i-1))

вывод результатов в файл

Текст программы:

```
/*============================================================*/
/* Программа "myscheme1.c"                                    */
/*============================================================*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <math.h>
#include "mycom.h"

double xa, xb, xc;

char sname[128];
char vname[12] = "uu";

double u0(double x);
double u0(double x){
  if ((x>=xa) && (x<=xc)) return sin(pi*x);
  else return  0;
```

```
}

int main(void)
{
  int i, Nx=100, ntv, ntvmax;
  double a=1.0;
  double Lx, hx, tau, tv, tvmax, gam;
  double *xx, *uu, *vv;
  FILE *fp;

if ((fp = fopen("myscheme1.res", "w"))  == NULL)
  {
    printf("File not open\n");
    return 0;
  }

  xa = 0.0;
  xb = 10.0;
  xc = 1.0;
  Lx = xb-xa;
  hx = Lx/Nx;
  gam = 0.5;
  tau = gam * (hx / a);

  ntv = 0;
  tv = 0;
  tvmax = 2.0;
  ntvmax = (int)(tvmax/tau);

  xx = (double*)(malloc(sizeof(double)*(Nx+1)));
  uu = (double*)(malloc(sizeof(double)*(Nx+1)));
  vv = (double*)(malloc(sizeof(double)*(Nx+1)));

  fprintf(fp,"xa=%le\n",xa);
  fprintf(fp,"xb=%le\n",xb);
  fprintf(fp,"xc=%le\n",xc);
  fprintf(fp,"Lx=%le\n",Lx);
  fprintf(fp,"Nx=%d\n",Nx);
  fprintf(fp,"hx=%le\n",hx);
  fprintf(fp,"a=%le\n",a);
  fprintf(fp,"tau=%le\n",tau);
  fprintf(fp,"gam=%le\n",gam);
  fprintf(fp,"tvmax=%le\n",tvmax);
  fprintf(fp,"ntvmax=%d\n",ntvmax);

  printf("Nx=%d\n",Nx);
  printf("ntvmax=%d\n",ntvmax);

  fprintf(fp,"\nntv=%d tv=%le\n",ntv,tv);

  for (i=0; i<=Nx; i++){
    xx[i] = xa + hx * i;
    uu[i] = u0(xx[i]);
    if (fabs(uu[i])>1e-4)
      fprintf(fp,"i=%d x=%le u0=%le\n", i, xx[i], uu[i]);
  }

  do {
    ntv++;
    tv += tau;

    printf("ntv=%d tv=%le\n",ntv,tv);
    fprintf(fp,"\nntv=%d tv=%le\n",ntv,tv);

    vv[0] = 0.0;
    for (i=1; i<=Nx; i++) {
      vv[i] = uu[i] - gam * (uu[i] - uu[i-1]);
    }
```

```
    for (i=0; i<=Nx; i++) uu[i] = vv[i];

    for (i=0; i<=Nx; i++)
      if (fabs(uu[i])>1e-4)
        fprintf(fp,"i=%d x=%le uu=%le\n", i, xx[i], uu[i]);

    {
      FILE *F;
      for (i=0; i<128; i++) sname[i] = 0;
      sprintf(sname,"myscheme1_%s_%06d.dat",vname,ntv);
      F = fopen(sname, "wt");
//    fprintf(F,"TITLE = \"FUNC\"\n");
//    fprintf(F,"VARIABLES = \"X\", \"F\"\n");
//    fprintf(F,"ZONE I=%d, F=POINT\n",Nx+1);
      for (i=0; i<=Nx; i++)
        fprintf(F,"%20.13le %20.13le\n",xx[i],uu[i]);
      fclose(F);
    }

  } while (tv<=tvmax);

  fclose(fp);
  return 0;
}
```

Трансляция:

```
> gcc -o myscheme1 -O2 myscheme1.c -lm
```

Выполнение:

```
>myscheme1
ntv=1 tv=5.000000e-02
ntv=2 tv=1.000000e-01
ntv=3 tv=1.500000e-01
...
ntv=38 tv=1.900000e+00
ntv=39 tv=1.950000e+00
ntv=40 tv=2.000000e+00

>ls -l myscheme1*
-rwxr-xr-x 1 mephi99 mephi 16944 Oct  9 23:53 myscheme1
-rw-r--r-- 1 mephi99 mephi  1999 Oct  9 23:53 myscheme1.c
-rw-r--r-- 1 mephi99 mephi 76709 Oct  9 23:53 myscheme1.res
-rw-r--r-- 1 mephi99 mephi  4298 Oct  9 23:53 myscheme1_uu_000001.dat
-rw-r--r-- 1 mephi99 mephi  4298 Oct  9 23:53 myscheme1_uu_000002.dat
-rw-r--r-- 1 mephi99 mephi  4298 Oct  9 23:53 myscheme1_uu_000003.dat
...
-rw-r--r-- 1 mephi99 mephi  4298 Oct  9 23:53 myscheme1_uu_000038.dat
-rw-r--r-- 1 mephi99 mephi  4298 Oct  9 23:53 myscheme1_uu_000039.dat
-rw-r--r-- 1 mephi99 mephi  4298 Oct  9 23:53 myscheme1_uu_000040.dat
```

Визуальный анализ с помощью пакета gnuplot:

```
>myscheme1.gpl
>gnuplot myscheme1.dem
...
```

Задание: написать программу для схемы (Б).