Семинар 12. Решение двумерного уравнения Пуассона итерационными методами.

1. Постановка задачи.

$$\frac{\partial}{\partial x_1}\left(k_1(x_1,x_2)\frac{\partial u}{\partial x_1}\right)+\frac{\partial}{\partial x_2}\left(k_2(x_1,x_2)\frac{\partial u}{\partial x_2}\right)=-f(x_1,x_2),\quad (x_1,x_2)\in D=(0,1)\times(0,1),$$

$$u(x_1,x_2)=g(x_1,x_2),\quad (x_1,x_2)\in\partial D.$$

2. Конечно-разностная схема.

Равномерная сетка: $\Omega=\omega_{x_1}\times\omega_{x_2},\ \omega_{x_\alpha}=\left\{x_\alpha=i_\alpha h_\alpha,\ i_\alpha=0,...,N_\alpha,\ h_\alpha=1/N_\alpha\right\},\ \alpha=1,2.$

Разностная схема:

$$\frac{1}{\hbar_1}\left\{k_{1,i_1+1/2,i_2}\frac{y_{i_1+1,i_2}-y_{i_1,i_2}}{h_1}-k_{1,i_1-1/2,i_2}\frac{y_{i_1,i_2}-y_{i_1-1,i_2}}{h_1}\right\}+$$

$$+\frac{1}{\hbar_2}\left\{k_{2,i_1,i_2+1/2}\frac{y_{i_1,i_2+1}-y_{i_1,i_2}}{h_2}-k_{2,i_1,i_2-1/2}\frac{y_{i_1,i_2}-y_{i_1,i_2-1}}{h_2}\right\}=-f_{i_1,i_2},\quad i_\alpha=1,...,N_\alpha-1,$$

$$y_{i_1,i_2}=g_{i_1,i_2},\quad i_\alpha=0,N_\alpha,\quad \alpha=1,2.$$

Эквивалентные уравнения:

$$\left(\frac{\hbar_2}{h_1}\left(k_{1,i_1+1/2,i_2}+k_{1,i_1-1/2,i_2}\right)+\frac{\hbar_1}{h_2}\left(k_{2,i_1,i_2+1/2}+k_{2,i_1,i_2-1/2}\right)\right)y_{i_1,i_2}-\frac{\hbar_2}{h_1}k_{1,i_1+1/2,i_2}y_{i_1+1,i_2}-\frac{\hbar_2}{h_1}k_{1,i_1-1/2,i_2}y_{i_1-1,i_2}-$$

$$-\frac{\hbar_1}{h_2}k_{2,i_1,i_2+1/2}y_{i_1,i_2+1}-\frac{\hbar_1}{h_2}k_{2,i_1,i_2-1/2}y_{i_1,i_2-1}=\hbar_1\hbar_2 f_{i_1,i_2},\quad i_\alpha=1,...,N_\alpha-1,$$

$$y_{i_1,i_2}=g_{i_1,i_2},\quad i_\alpha=0,N_\alpha,\quad \alpha=1,2.$$

3. Решение уравнения Пуассона итерационными методами.

3.1. Метод простой итерации.

$$\overset{s+1}{y}_{i_1,i_2}=\overset{s}{y}_{i_1,i_2}-\tau\left(\frac{\hbar_2}{h_1}\left(k_{1,i_1+1/2,i_2}+k_{1,i_1-1/2,i_2}\right)+\frac{\hbar_1}{h_2}\left(k_{2,i_1,i_2+1/2}+k_{2,i_1,i_2-1/2}\right)\right)\overset{s}{y}_{i_1,i_2}+$$

$$\tau\frac{\hbar_2}{h_1}k_{1,i_1+1/2,i_2}\overset{s}{y}_{i_1+1,i_2}+\tau\frac{\hbar_2}{h_1}k_{1,i_1-1/2,i_2}\overset{s}{y}_{i_1-1,i_2}+\tau\frac{\hbar_1}{h_2}k_{2,i_1,i_2+1/2}\overset{s}{y}_{i_1,i_2+1}+\tau\frac{\hbar_1}{h_2}k_{2,i_1,i_2-1/2}\overset{s}{y}_{i_1,i_2-1}+\tau\hbar_1\hbar_2 f_{i_1,i_2},\quad s=0,1,...$$

Скорость сходимости $\left\|r^n\right\|\le q^n\left\|r^0\right\|,\quad q=\max\limits_{\lambda_{\min}\le\lambda\le\lambda_{\max}}\left|1-\tau\lambda\right|.$

Выбор параметра: $0<\tau<\dfrac{2}{\lambda_{\max}},\quad \tau=\tau_0=\dfrac{2}{\lambda_{\max}+\lambda_{\min}},\quad \lambda_{\min}=\min\lambda(A),\quad \lambda_{\max}=\max\lambda(A).$

Для достижения точности ε потребуется $n(\varepsilon)=1+\left\lfloor \ln\varepsilon^{-1}/\ln q^{-1}\right\rfloor$ итераций, $q=1-\tau\lambda_{\min}$. Для оптимального параметра получаем $q=q_0=1-\tau_0\lambda_{\min}=1-\dfrac{2}{1+\mu}=\dfrac{\mu-1}{\mu+1},\quad n(\varepsilon)=1+\left\lfloor \ln\varepsilon^{-1}/\ln\dfrac{\mu+1}{\mu-1}\right\rfloor$. Здесь $\mu=\dfrac{\lambda_{\max}}{\lambda_{\min}}$ – число обусловленности.

3.2. Метод простой итерации с ускорением по Чебышеву.

Возьмем нестационарный метод, а именно различные параметры $\{\tau_s\}$. Тогда скорость сходимости выражается формулами: $\left\|r^n\right\|\le q\left\|r^0\right\|,\quad q=\max\limits_{\lambda_{\min}\le\lambda\le\lambda_{\max}}\left|\prod\limits_{s=0}^{n-1}(1-\tau_{s+1}\lambda)\right|$. Если выбрать Чебышевский набор итерационных параметров:

$$\tau=t_{s+1}=\left[\frac{\lambda_{\max}+\lambda_{\min}}{2}+\frac{\lambda_{\max}-\lambda_{\min}}{2}\cos\frac{\pi(2(s+1)-1)}{2n}\right]^{-1}=\frac{\tau_0}{1+\dfrac{\mu-1}{\mu+1}\cos\dfrac{\pi(2(s+1)-1)}{2n}},\quad s=0,...,n-1,$$

то получим максимальную скорость сходимости. Здесь n – длина итерационной серии. Для достижения точности ε необходимо сделать $n(\varepsilon)=1+\left\lfloor 0.5\sqrt{\mu}\ln\varepsilon^{-1}\right\rfloor$. Для реализации метода важен порядок использования параметров. В случае, когда $n=2^m$, этот порядок вычисляется относительно просто: $\tau_{2s}=t_{i(s)},\quad \tau_{2s+1}=t_{n-1-i(s)},\quad s=0,...,n/2-1,\quad i(s)=...$

3.3. Трехслойный метод Чебышева.

$$y^1=\left(E-\tau_0A\right)y^0+\tau_0 f,\quad y^{s+1}=\alpha_{s+1}\left(E-\tau_0A\right)y^s+(1-\alpha_{s+1})y^{s-1}+\tau_0\alpha_{s+1}f,\quad s=1,2,...,$$

$$\alpha_1=2,\quad \alpha_{s+1}=\frac{4}{4-\rho^2\alpha_s},\quad \rho=\frac{\mu-1}{\mu+1}.$$

Скорость сходимости $\left\|r^n\right\|\le\frac{2q^n}{1+q^{2n}}\left\|r^0\right\|,\quad q=\frac{\sqrt{\mu}-1}{\sqrt{\mu}+1}.$

3.4. Метод Якоби.

Каноническая двухслойная схема итераций:

$$B_{s+1}\frac{\overset{s+1}{y}-\overset{s}{y}}{\tau_{s+1}}+A\overset{s}{y}=f,\quad s=0,1,....$$

Метод Якоби: B_s – диагональные матрицы. Обычно рассматривают стационарный метод с $B_s=D_A$.

Для нашей задачи имеем:

$$\overset{s+1}{y}_{i_1,i_2}=(1-\tau)\overset{s}{y}_{i_1,i_2}+\tau\left(\frac{\hbar_2}{h_1}\left(k_{1,i_1+1/2,i_2}+k_{1,i_1-1/2,i_2}\right)+\frac{\hbar_1}{h_2}\left(k_{2,i_1,i_2+1/2}+k_{2,i_1,i_2-1/2}\right)\right)^{-1}\times$$

$$\times\left(\frac{\hbar_2}{h_1}k_{1,i_1+1/2,i_2}\overset{s}{y}_{i_1+1,i_2}+\frac{\hbar_2}{h_1}k_{1,i_1-1/2,i_2}\overset{s}{y}_{i_1-1,i_2}+\frac{\hbar_1}{h_2}k_{2,i_1,i_2+1/2}\overset{s}{y}_{i_1,i_2+1}+\frac{\hbar_1}{h_2}k_{2,i_1,i_2-1/2}\overset{s}{y}_{i_1,i_2-1}+\hbar_1\hbar_2 f_{i_1,i_2}\right),\quad s=0,1,...$$

Выбор параметров: либо один оптимальный $\tau=\tau_0=\frac{2}{\min\lambda(D^{-1}A)+\max\lambda(D^{-1}A)}$, либо чебышевский набор. Возможно объединить метод Якоби и трехслойный метод Чебышева, если решать систему $D_A^{-1}Ay=D_A^{-1}f$.

3.5. Метод Зейделя–Некрасова–Якоби.

В данном методе $B_s=E+D_A^{-1}A_-$:

$$\overset{s+1}{y}_{i_1,i_2}=(1-\tau)\overset{s}{y}_{i_1,i_2}+\tau\left(\frac{\hbar_2}{h_1}\left(k_{1,i_1+1/2,i_2}+k_{1,i_1-1/2,i_2}\right)+\frac{\hbar_1}{h_2}\left(k_{2,i_1,i_2+1/2}+k_{2,i_1,i_2-1/2}\right)\right)^{-1}\times$$

$$\times\left(\frac{\hbar_2}{h_1}k_{1,i_1+1/2,i_2}\overset{s}{y}_{i_1+1,i_2}+\frac{\hbar_2}{h_1}k_{1,i_1-1/2,i_2}\overset{s+1}{y}_{i_1-1,i_2}+\frac{\hbar_1}{h_2}k_{2,i_1,i_2+1/2}\overset{s}{y}_{i_1,i_2+1}+\frac{\hbar_1}{h_2}k_{2,i_1,i_2-1/2}\overset{s+1}{y}_{i_1,i_2-1}+\hbar_1\hbar_2 f_{i_1,i_2}\right),\quad s=0,1,...$$

Выбор параметров τ_s: можно выбрать также, как и выше.

4. Параллельная реализация на примерах.

Тестовая задача:

$$k_\alpha\equiv1,\quad f(x_1,x_2)=2\pi^2\sin(\pi x_1)\sin(\pi x_2),\quad g(x_1,x_2)\equiv0,\quad u(x_1,x_2)=\sin(\pi x_1)\sin(\pi x_2).$$

4.1. Метод Якоби.

$$\overset{s+1}{y}_{i_1,i_2}=(1-\tau)\overset{s}{y}_{i_1,i_2}+\frac{\tau}{2}\left(\frac{h_2}{h_1}+\frac{h_1}{h_2}\right)^{-1}\left(\frac{h_2}{h_1}\left(\overset{s}{y}_{i_1+1,i_2}+\overset{s}{y}_{i_1-1,i_2}\right)+\frac{h_1}{h_2}\left(\overset{s}{y}_{i_1,i_2+1}+\overset{s}{y}_{i_1,i_2-1}\right)+h_1h_2f_{i_1,i_2}\right),\quad s=0,1,...,$$

$$\overset{s+1}{y}_{i_1,i_2}=0,\quad i_\alpha=0,N_\alpha,\quad \alpha=1,2.$$

Выбор оптимального параметра:

$$\lambda_{\min}=\lambda_{\min}(D^{-1}A)=\frac{1}{2}\left(\frac{h_2}{h_1}+\frac{h_1}{h_2}\right)^{-1}h_1h_2\left(\frac{4}{h_1^2}\sin^2\frac{\pi h_1}{2}+\frac{4}{h_2^2}\sin^2\frac{\pi h_2}{2}\right)\approx\frac{\pi^2h_1^2h_2^2}{h_1^2+h_2^2},$$

$$\lambda_{\max}=\lambda_{\max}(D^{-1}A)=\frac{1}{2}\left(\frac{h_2}{h_1}+\frac{h_1}{h_2}\right)^{-1}h_1h_2\left(\frac{4}{h_1^2}\cos^2\frac{\pi h_1}{2}+\frac{4}{h_2^2}\cos^2\frac{\pi h_2}{2}\right)\approx 2,$$

$$\tau=\tau_0=\frac{2}{\lambda_{\min}+\lambda_{\max}}\approx\frac{2}{\dfrac{\pi^2h_1^2h_2^2}{h_1^2+h_2^2}+2}.$$

Если шаги одинаковы, то $\tau_0\approx\dfrac{1}{1+0.25\pi^2h^2}$.

Параллельная реализация проводится на решетке процессоров.

4.2. Метод Зейделя–Некрасова–Якоби.

$$\overset{s+1}{y}_{i_1,i_2}=(1-\tau)\overset{s}{y}_{i_1,i_2}+\frac{\tau}{2}\left(\frac{h_2}{h_1}+\frac{h_1}{h_2}\right)^{-1}\left(\frac{h_2}{h_1}\left(\overset{s}{y}_{i_1+1,i_2}+\overset{s+1}{y}_{i_1-1,i_2}\right)+\frac{h_1}{h_2}\left(\overset{s}{y}_{i_1,i_2+1}+\overset{s+1}{y}_{i_1,i_2-1}\right)+h_1h_2f_{i_1,i_2}\right),\quad s=0,1,...,$$

$$\overset{s+1}{y}_{i_1,i_2}=0,\quad i_\alpha=0,N_\alpha,\quad \alpha=1,2.$$

5. Реализация примеров.

Пример 1. Метод Якоби с ЧНП (ex16a.c).

```
#include <stdio.h> #include <stdlib.h> #include <string.h> #include <unistd.h>
#include <math.h> #include "mycom.h" #include "mynet.h" #include "myrand.h"
#include "myio.h"
int np, mp, nl, ier, lp;
int np1, np2, mp1, mp2;
int mp_l, mp_r, mp_b, mp_t;
char pname[MPI_MAX_PROCESSOR_NAME];
int lname; char vname[1024]; char sname[1024]; union_t buf;
double tick, t1, t2, t3;
FILE *Fi = NULL; FILE *Fo = NULL;
int n1, n2, im, nc, mc, itm;
double pi2, eps;
double f(double x1, double x2); double f(double x1, double x2) {
  return pi2*dsin(pi*x1)*dsin(pi*x2); }
double u(double x1, double x2); double u(double x1, double x2) {
  return dsin(pi*x1)*dsin(pi*x2); }
int main(int argc, char *argv[])
{
  int i, n, m, i1, i2, k1, k2, n1p, n2p, n12p, it;
  int i11, i12, i21, i22, nc1, nc2, nc1m, nc2m, nc12;
  double h1, h2, h12, tau, tau0, g12, g21, rka, dka;
  double s0, s1, s2, s3, s1m, s1p, s2m, s2p;
  double *rr, *xx1, *xx2, *ff, *yy0, *yy1;
  double *rr_l, *ss_l, *rr_r, *ss_r, *rr_b, *ss_b, *rr_t, *ss_t;
  MyNetInit(&argc,&argv,&np,&mp,&nl,pname,&tick);
  fprintf(stderr,"Netsize: %d, process: %d, system: %s, tick=%12le\n",np,mp,pname,tick);
  sleep(1);
  if (argc>1) strcpy(vname,argv[1]); else strcpy(vname,"ex16a"); lname = strlen(vname);
  if (mp==0) fprintf(stderr,"Base name is %s\n",vname);
  sprintf(sname,"%s.p%02d",vname,mp);
  ier = fopen_m(&Fo,sname,"wt");
  if (ier!=0) mpierr("Protocol file not opened",1);
  fprintf(Fo,"Netsize: %d, process: %d, system: %s, tick=%12le\n",np,mp,pname,tick);
  if (mp==0) {
    sprintf(sname,"%s.d",vname);
    ier = fopen_m(&Fi,sname,"rt");
    if (ier!=0) mpierr("Data file not opened",2);
    fscanf(Fi,"n1=%d\n",&n1);  fscanf(Fi,"n2=%d\n",&n2);
    fscanf(Fi,"im=%d\n",&im);  fscanf(Fi,"lp=%d\n",&lp);
    fclose_m(&Fi);
```

```
    if (argc>2) sscanf(argv[2],"%d",&n1); if (argc>3) sscanf(argv[3],"%d",&n2);
    if (argc>4) sscanf(argv[4],"%d",&im); if (argc>5) sscanf(argv[5],"%d",&lp);
  }
  if (np>1) {
    if (mp==0) {
      buf.idata[0] = n1; buf.idata[1] = n2; buf.idata[2] = im; buf.idata[3] = lp;
    }
    MPI_Bcast(buf.ddata,2,MPI_DOUBLE,0,MPI_COMM_WORLD);
    if (mp>0) {
      n1 = buf.idata[0]; n2 = buf.idata[1]; im = buf.idata[2]; lp = buf.idata[3];
    }
  }
  pi2 = 2.0*pi*pi;
  n1p = n1+1; n2p = n2+1; n12p = n1p*n2p; h1 = 1.0/n1; h2 = 1.0/n2;
  s0 = dmin(h1,h2); eps = pow(s0,3); eps = dmax(eps,1e-14);
  s0 = log(1.0/eps)*n12p; if (s0>2000000000) itm = 2000000000; else itm = (int)s0;
  tau0 = 0.25; if (argc>6) sscanf(argv[6],"%le",&tau0); tau = tau0;
  mc = (int)(0.5*log(1.0*n12p)/log(2.0)); if (argc>7) sscanf(argv[7],"%d",&mc);
  if (mc<1){ mc=0; nc=1; } else nc = 1 << mc;
  g12 = h1/h2; g21 = h2/h1; s0 = g12+g21;
  g12 = 0.5*g12/s0; g21 = 0.5*g21/s0; h12 = 0.5*h1*h2/s0;
  My2DGrid(np,mp,n1,n2,&np1,&np2,&mp1,&mp2);
  if (mp1 ==     0) mp_l = -1; else mp_l = mp - 1;
  if (mp1 == np1-1) mp_r = -1; else mp_r = mp + 1;
  if (mp2 ==     0) mp_b = -1; else mp_b = mp - np1;
  if (mp2 == np2-1) mp_t = -1; else mp_t = mp + np1;
  MyRange(np1,mp1,0,n1,&i11,&i12,&nc1); nc1m = nc1 - 1;
  MyRange(np2,mp2,0,n2,&i21,&i22,&nc2); nc2m = nc2 - 1; nc12 = nc1 * nc2;
  fprintf(Fo,"n1=%d n2=%d mc=%d nc=%d\n",n1,n2,mc,nc);
  fprintf(Fo,"n12=%d itm=%d h1=%le h2=%le eps=%le\n",n12p,itm,h1,h2,eps);
  fprintf(Fo,"Grid=%dx%d coords=(%d,%d)\n",np1,np2,mp1,mp2);
  fprintf(Fo,"mp_l=%d mp_r=%d mp_b=%d mp_t=%d\n",mp_l,mp_r,mp_b,mp_t);
  fprintf(Fo,"i11=%d i12=%d nc1=%d\n",i11,i12,nc1);
  fprintf(Fo,"i21=%d i22=%d nc2=%d\n",i21,i22,nc2);
  if (mp==0){
    fprintf(stderr,"n1=%d n2=%d mc=%d nc=%d n12=%d itm=%d\n",n1,n2,mc,nc,n12p,itm);
    fprintf(stderr,"grid=%dx%d h1=%10.3le h2=%10.3le eps=%10.3le\n",np1,np2,h1,h2,eps);
  }
  rr = (double*)(malloc(sizeof(double)*nc)); mychebset(mc,nc,rr);
  if (lp>0) for (m=0; m<nc; m++) fprintf(Fo,"m=%d rr=%le\n",m,rr[m]);
  t1 = MPI_Wtime();
  xx1 = (double*)(malloc(sizeof(double)*nc1));
  xx2 = (double*)(malloc(sizeof(double)*nc2));
  ff  = (double*)(malloc(sizeof(double)*nc12));
  yy0 = (double*)(malloc(sizeof(double)*nc12));
  yy1 = (double*)(malloc(sizeof(double)*nc12));
  if (mp_l>=0) {
    rr_l = (double*)(malloc(sizeof(double)*nc2));
    ss_l = (double*)(malloc(sizeof(double)*nc2));
  }
  if (mp_r>=0) {
    rr_r = (double*)(malloc(sizeof(double)*nc2));
    ss_r = (double*)(malloc(sizeof(double)*nc2));
  }
  if (mp_b>=0) {
    rr_b = (double*)(malloc(sizeof(double)*nc1));
    ss_b = (double*)(malloc(sizeof(double)*nc1));
  }
  if (mp_t>=0) {
    rr_t = (double*)(malloc(sizeof(double)*nc1));
    ss_t = (double*)(malloc(sizeof(double)*nc1));
  }
  for (i1=0; i1<nc1; i1++) xx1[i1] = h1 * (i11+i1);
  for (i2=0; i2<nc2; i2++) xx2[i2] = h2 * (i21+i2);
// Initialization of iterations:
  for (m=0; m<nc12; m++)  ff[m] = 0.0;
  for (m=0; m<nc12; m++) yy0[m] = 0.0;
  for (m=0; m<nc12; m++) yy1[m] = 0.0;
```

```
  for (k2=0; k2<nc2; k2++) {
    i2 = i21 + k2;
    if ((i2>0) && (i2<n2)) {
      for (k1=0; k1<nc1; k1++) {
        i1 = i11 + k1;
        if ((i1>0) && (i1<n1)) {
          m = nc1 * k2 + k1; ff[m] = h12 * f(xx1[k1],xx2[k2]); yy1[m] = ff[m];
        }
      }
    }
  }
  if (lp>0) {
    for (k2=0; k2<nc2; k2++) {
      i2 = i21 + k2;
      for (k1=0; k1<nc1; k1++) {
        i1 = i11 + k1; m = nc1 * k2 + k1;
        fprintf(Fo,"i1=%d i2=%d x1=%le x2=%le f=%le\n",i1,i2,xx1[k1],xx2[k2],ff[m]);
      }
    }
  }
  it = 0;
// Iterative loop:
  do {
    if (mc>0){ // Chebyshov's speedup
      m = it % nc;  tau = tau0*rr[m];
    }
    for (m=0; m<nc12; m++) yy0[m] = yy1[m]; // update
    if (np>1) {
      if (mp_l>=0) {
        i1 = 0; for (i2=0; i2<nc2; i2++) { m = nc1 * i2 + i1; ss_l[i2] = yy0[m]; }
      }
      if (mp_r>=0) {
        i1 = nc1m; for (i2=0; i2<nc2; i2++) { m = nc1 * i2 + i1; ss_r[i2] = yy0[m]; }
      }
      if (mp_b>=0) {
        i2 = 0; for (i1=0; i1<nc1; i1++) { m = nc1 * i2 + i1; ss_b[i1] = yy0[m]; }
      }
      if (mp_t>=0) {
        i2 = nc2m; for (i1=0; i1<nc1; i1++) { m = nc1 * i2 + i1; ss_t[i1] = yy0[m]; }
      }
      BndAExch2D(mp_l,nc2,ss_l,rr_l,
                 mp_r,nc2,ss_r,rr_r,
                 mp_b,nc1,ss_b,rr_b,
                 mp_t,nc1,ss_t,rr_t);
    }
    rka = 0.0;
    for (k2=0; k2<nc2; k2++) {
      i2 = i21 + k2;
      if ((i2>0) && (i2<n2)) {
        for (k1=0; k1<nc1; k1++) {
          i1 = i11 + k1;
          if ((i1>0) && (i1<n1)) {
            m = nc1 * k2 + k1; s0 = yy0[m];
            if (k1==   0) {if(mp_l>=0) s1m = rr_l[k2]; else s1m =0.0;}
            else s1m = yy0[m-1];
            if (k1==nc1m) {if(mp_r>=0) s1p = rr_r[k2]; else s1p =0.0;}
            else s1p = yy0[m+1];
            if (k2==   0) {if(mp_b>=0) s2m = rr_b[k1]; else s2m =0.0;}
            else s2m = yy0[m-nc1];
            if (k2==nc2m) {if(mp_t>=0) s2p = rr_t[k1]; else s2p =0.0;}
            else s2p = yy0[m+nc1];
            s1 = g21 * (s1m + s1p); s2 = g12 * (s2m + s2p); s3 = ff[m];
            yy1[m] = s1 + s2 + s3 - s0; // residual
            s0 = dabs(yy1[m]); rka = dmax(rka,s0);
          }
        }
      }
    }
```

```
    if (np>1) {
      s0 = rka; MPI_Allreduce(&s0,&rka,1,MPI_DOUBLE,MPI_MAX,MPI_COMM_WORLD);
    }
    if ((lp>0) && (mp==0)) fprintf(stderr,"it=%d tau=%le rka=%le\n",it,tau,rka);
    if (lp>1) {
      fprintf(Fo,"it=%d tau=%le rka=%le\n",it,tau,rka);
      for (k2=0; k2<nc2; k2++) {
        i2 = i21 + k2;
        for (k1=0; k1<nc1; k1++) {
          i1 = i11 + k1; m = nc1 * k2 + k1;
          fprintf(Fo,"i1=%8d i2=%8d y0=%le\n",i1,i2,yy0[m]);
        }
      }
    }
    if ((rka<=eps) || (it>=itm)) {
      for (m=0; m<nc12; m++) yy1[m] = yy0[m];
      break;
    }
    it = it + 1;
    for (k2=0; k2<nc2; k2++) {
      i2 = i21 + k2;
      if ((i2>0) && (i2<n2)) {
        for (k1=0; k1<nc1; k1++) {
          i1 = i11 + k1;
          if ((i1>0) && (i1<n1)) {
            m = nc1 * k2 + k1; yy1[m] = yy0[m] + tau * yy1[m];
          }
        }
      }
    }
  } while (it<=itm);
  dka = 0.0;
  for (k2=0; k2<nc2; k2++) {
    i2 = i21 + k2;
    for (k1=0; k1<nc1; k1++) {
      i1 = i11 + k1; m = nc1 * k2 + k1;
      s0 = u(xx1[k1],xx2[k2]); s1 = dabs(yy1[m]-s0); dka = dmax(dka,s1);
      if (lp>0) fprintf(Fo,"i1=%8d i2=%8d y=%le u=%le d=%le\n",i1,i2,yy1[m],s0,s1);
    }
  }
  if (np>1) {
    s0 = dka; MPI_Allreduce(&s0,&dka,1,MPI_DOUBLE,MPI_MAX,MPI_COMM_WORLD);
  }
  t1 = MPI_Wtime() - t1;
  fprintf(Fo,"it=%d rka=%le dka=%le time=%le\n",it,rka,dka,t1);
  if (mp==0)
    fprintf(stderr,"np=%d (%dx%d) n1=%8d n2=%8d it=%d rka=%le dka=%le time=%le\n",
      np,np1,np2,n1,n2,it,rka,dka,t1);
  sprintf(sname,"%s_%02d.dat",vname,np);
  OutFun2DP(sname,np,mp,nc1,nc2,xx1,xx2,yy1);
  MPI_Finalize();
  return 0;
}
```

Трансляция:

```
>mpicc -o ex16a.px -O2 ex16a.c mycom.c mynet.c myio.c myrand.c -lm
```

Результаты расчетов:

```
>mpirun -np <1-24> -nolocal -machinefile hosts ex16a.px a <2-1024> <2-1024>   (в классе)
>mpirun -np <1-24> ex16a.px a <2-1024> <2-1024>                                (на сервере)
np= 1  (1x1) n1=     2 n2=     2 it=   0 rka=0.000000e+00 dka=2.337006e-01 time=5.000000e-06
np= 1  (1x1) n1=     4 n2=     4 it=   4 rka=6.664829e-03 dka=3.027414e-02 time=1.300000e-05
np= 1  (1x1) n1=     8 n2=     8 it=   8 rka=5.405428e-04 dka=5.849598e-03 time=5.000000e-05
np= 1  (1x1) n1=    16 n2=    16 it=  16 rka=3.580093e-05 dka=1.355761e-03 time=2.450000e-04
np= 1  (1x1) n1=    32 n2=    32 it=  32 rka=2.269387e-06 dka=3.322884e-04 time=1.463000e-03
np= 1  (1x1) n1=    64 n2=    64 it=  64 rka=1.423352e-07 dka=8.265655e-05 time=1.002400e-02
np= 1  (1x1) n1=   128 n2=   128 it= 128 rka=8.903733e-09 dka=2.063817e-05 time=7.370100e-02
np= 1  (1x1) n1=   256 n2=   256 it= 256 rka=5.570938e-10 dka=5.157920e-06 time=4.412720e-01
np= 1  (1x1) n1=   512 n2=   512 it= 512 rka=3.844114e-11 dka=1.289379e-06 time=4.359555e+00
np= 1  (1x1) n1=  1024 n2=  1024 it=1024 rka=2.391924e-11 dka=3.223407e-07 time=3.397578e+01
```

```
np= 1   (1x1)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=4.359555e+00
np= 2   (1x2)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=2.509570e+00
np= 3   (1x3)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.853683e+00
np= 4   (2x2)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.575268e+00
np= 5   (1x5)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.697655e+00
np= 6   (2x3)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.410437e+00
np= 7   (1x7)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.550687e+00
np= 8   (2x4)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.259869e+00
np= 9   (3x3)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.630450e+00
np=10   (2x5)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.092398e+00
np=11  (1x11)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.283869e+00
np=12   (3x4)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.198221e+00
np=15   (3x5)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.473445e+00
np=16   (4x4)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.519445e+00
np=20   (4x5)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.520552e+00
np=24   (4x6)  n1=    512  n2=    512  it= 512  rka=3.844114e-11  dka=1.289379e-06  time=1.524282e+00

np= 1   (1x1)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=3.397578e+01
np= 2   (1x2)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=1.825585e+01
np= 3   (1x3)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=1.316699e+01
np= 4   (2x2)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=1.053883e+01
np= 5   (1x5)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=9.632603e+00
np= 6   (2x3)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=7.909441e+00
np= 7   (1x7)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=7.578117e+00
np= 8   (2x4)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=6.534893e+00
np= 9   (3x3)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=5.893631e+00
np=10   (2x5)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=6.094514e+00
np=11  (1x11)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=6.221716e+00
np=12   (3x4)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=5.829750e+00
np=13  (1x13)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=7.089344e+00
np=14   (2x7)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=6.674810e+00
np=15   (3x5)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=6.097073e+00
np=16   (4x4)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=6.121966e+00
np=20   (4x5)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=6.039219e+00
np=24   (4x6)  n1=   1024  n2=   1024  it=1024  rka=2.391924e-11  dka=3.223407e-07  time=5.609448e+00
```