Εθνικο & Καποδιστριακο Πανεπιστημιο Αθηνων
Τμημα Πληροφορικης & Τηλεπικοινωνιων, ΠΜΣ Υπολογιστικη Επιστημη

Μάθημα: *«Επιστημονικοί Υπολογισμοί»*
Διδάσκων: *Φίλιππος Τζαφέρης*
Χειμερινό Εξάμηνο 2006-2007
Άγγελος Μαντζαφλάρης, amantzaf@math.uoa.gr
Τρίτη 5 Δεκεμβρίου, 2006

# Επίλυση της Self-adjoint μορφής

$$\frac{\partial u}{\partial t} = \frac{\partial u}{\partial x}\left(a(x)\frac{\partial u}{\partial x}\right)$$

Σε αυτήν την εργασία παρουσιάζεται κώδικας σε γλώσσα C για την αριθμητική λύση της παραπάνω Self-adjoint διαφορικής εξίσωσης, λαμβάνοντας

$$a(x) = 1 \quad , \quad \frac{1}{1+x^{10}} \quad , \quad \frac{1}{1+x^{20}} \quad , \quad \frac{1}{1+x^{30}}$$

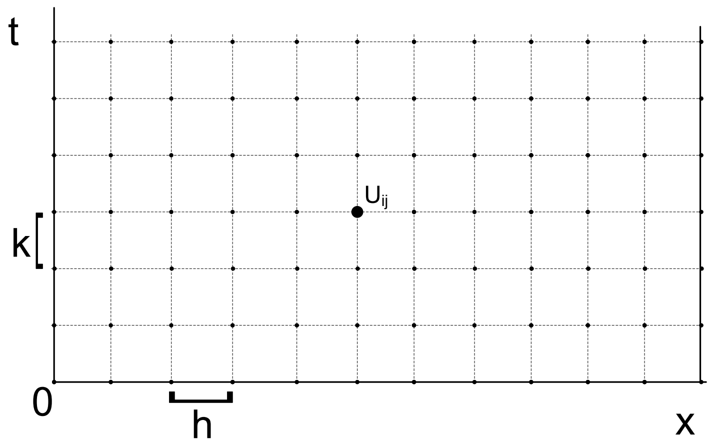
και αρχικές, συνοριακές συνθήκες

- Σταθερές:

$$u(x,0) = \frac{1}{2} \qquad , \qquad u(0,t) = u(1,t) = 0$$

- Με μερικές παραγώγους:

$$u(x,0) = 1 \quad , \quad \frac{\partial u}{\partial x} = u \ \text{ για } x = 0,\ t > 0 \quad , \quad \frac{\partial u}{\partial x} = -u \ \text{ για } x = 1,\ t > 0$$




Η συνάρτηση $a(x)$ δίνεται από τη ρουτίνα `fun_a.c` (για την ύψωση σε δύναμη χρησιμοποιείται η βοηθητική `power.c`). Όλες οι παρακάτω μέθοδοι μπορούν να τρέξουν με χρήση της `main.c`. Σε αυτήν χρησιμοποιούνται και οι `input.c` (για είσοδο παραμέτρων) και `printdata.c` (για εκτύπωση δειγματοληπτικών δεδομένων). Η δειγματοληψία γίνεται σε έναν $6 \times 6$ πίνακα με τη συνάρτηση `sample.c`.

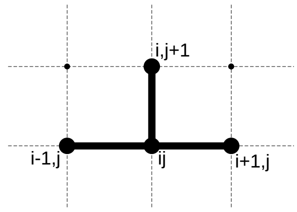
Τα εκτελέσιμα αρχεία της εργασίας είναι τρία:

1. `main.exe` Δίνει στο χρήστη τη δυνατότητα να εισάγει παραμέτρους και να τρέξει τις μεθόδους.
2. `test.exe` Εξάγει αρχείο με δειγματοληπτικά δεδομένα εκτελέσεων για διάφορες παραμέτρους.
3. `cn_conv.exe` Δίνει τις τιμές του $r$ για τις οποίες η μέθοδος Crank-Nicolson είναι αρκετά ευσταθής.

# 1 Άμεση μέθοδος

Η άμεση μέθοδος δίνεται από το σχήμα

$$U_{i,j+1} = ra_{i-\frac{1}{2}}U_{i-1,j} + [1 - r(a_{i-\frac{1}{2}} + a_{i-\frac{1}{2}})]U_{ij} + ra_{i+\frac{1}{2}}U_{i+1,j}$$




## 1.1 Σταθερές συνοριακές συνθήκες

```
void explct(int n, int m, double r, double k, double h, double **U,int fa)
//Epilish me thn Amesh me8odo
{
  int i,j,j0=(int)(STEP/k), j1=6*j0+1;
  double v1,v2;

  //Arxikes sin8ikes
  for (i=0;i<=n;i++) U[i][0]=0.5;
  //Sinoriakes Sin8ikes
  U[0][1]=0; U[n][1]=0;

  for (j=1;j<=m;j++)
  {   printf("[%d]\r",j);
      for (i=1;i<n;i++)
     {
      v1=fun_a(i+0.5,h,fa); v2=fun_a(i-0.5,h,fa);
      U[i][1] = r*v2*U[i-1][0] + ( 1-r*(v1+v2) )*U[i][0] + r*v1*U[i+1][0];
     }
  for (i=0;i<=n;i++) U[i][0]=U[i][1];

  if ( (j%j0==0) && (j<j1) ) sample(U,n,j,j0);
  }
}
```

## 1.2 Συνοριακές συνθήκες με μερικές παραγώγους

Οι συνοριακές συνθήκες δίνονται από τις εξισώσεις

$$U_{0,j+1} = [1 - 2r(1+h)]U_{0,j} + 2rU_{1,j}$$

$$U_{N,j+1} = [1 - 2r(1+h)]U_{N,j} + 2rU_{N-1,j}$$

```
void explct_ic(int n, int m, double r, double k, double h, double **U,int fa)
//Epilish me thn Amesh me8odo
{
  int i,j,j0=(int)(STEP/k), j1=6*j0+1;
  double v1,v2,v=1-2*r*(1+h);

  //Arxikes sin8ikes
  for (i=0;i<=n;i++) U[i][0]=1;

  for (j=1;j<=m;j++)
  {   printf("[%d]\r",j);
      //Sinoriakes Sin8ikes
      U[0][1]= v*U[0][0]+2*r*U[1][0];
      U[n][1]= v*U[n][0]+2*r*U[n-1][0];
      for (i=1;i<n;i++)
```

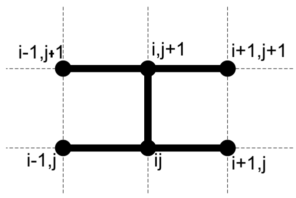
```
      {
       v1=fun_a(i+0.5,h,fa); v2=fun_a(i-0.5,h,fa);
       U[i][1] = r*v2*U[i-1][0] + ( 1-r*(v1+v2) )*U[i][0] + r*v1*U[i+1][0];
      }
  for (i=0;i<=n;i++) U[i][0]=U[i][1];

  if ( (j%j0==0) && (j<j1) ) sample(U,n,j,j0);
  }
}
```

# 2 Έμμεσες μέθοδοι

Το γενικό σχήμα

$$-\lambda r a_{i-\frac{1}{2}} U_{i-1,j+1} + [1+\lambda r(a_{i-\frac{1}{2}} + a_{i+\frac{1}{2}})]U_{i,j+1} - \lambda r a_{i+\frac{1}{2}} U_{i+1,j+1} =$$
$$= (1-\lambda) r a_{i-\frac{1}{2}} U_{i-1,j} + [1-(1-\lambda) r(a_{i-\frac{1}{2}} + a_{i+\frac{1}{2}})]U_{ij} + (1-\lambda) r a_{i+\frac{1}{2}} U_{i+1,j}$$




μας δίνει τη μέθοδο Crank-Nicolson για $\lambda = \frac{1}{2}$ και τη μέθοδο Douglas για $\lambda = \frac{1}{2} - \frac{1}{12r}$.

Η επίλυση τριδιαγώνιου συστήματος γίνεται με τη ρουτίνα `thomas.c`.

## 2.1 Σταθερές συνοριακές συνθήκες

```
void implicit(int n, int m, double r, double k, double h, double **U,int fa,double l)
//Emmeses me8odoi me sta8eres arxikes sin8ikes
// Gia l=0.5 einai h Crank Nicolson
// Gia l=1/2-1/(12r) einai h Crank Nicolson
{
     int i,j,t=0, j0=(int)(STEP/k) , j1=6*j0+1;
     double a[n],b[n],c[n],d[n], v1,v2;

     //Arxikes sin8ikes
     for (i=0;i<=n;i++) U[i][0]=0.5;
     //Sinoriakes Sin8ikes
     U[0][1]=0; U[n][1]=0;

     for (j=1;j<=m;j++)
     {   printf("[%d]\r",j);
         for (i=1;i<n;i++)
         {
             v1=fun_a(i+0.5,h,fa); v2=fun_a(i-0.5,h,fa);
             a[i]=-l*r*v2;
             b[i]=1+l*r*(v1+v2);
             c[i]=-l*r*v1;
             d[i]=(1-l)*r*v2*U[i-1][0] + (1-(1-l)*r*(v1+v2))*U[i][0]
             + (1-l)*r*v1*U[i+1][0];
         }
         d[1]  -=a[1]*U[0][1];
         d[n-1]-=c[n-1]*U[n][1];

     thomas(n,U,a,b,c,d);
     for (i=0;i<=n;i++) U[i][0]=U[i][1];
     if ( (j%j0==0) && (j<j1) ) sample(U,n,j,j0);
     }
}
```

## 2.2 Συνοριακές συνθήκες με μερικές παραγώγους

Οι συνοριακές συνθήκες δίνονται όμοια με πριν:

$$U_{0,j+1} = [1 - 2r(1+h)]U_{0,j} + 2rU_{1,j}$$

$$U_{N,j+1} = [1 - 2r(1+h)]U_{N,j} + 2rU_{N-1,j}$$

```
void implicit_ic(int n, int m, double r, double k, double h, double **U,int fa,double l)
//Emmeses me8odoi me arxikes sin8ikes me merikes paragogous
// Gia l=0.5 einai h Crank Nicolson
// Gia l=1/2-1/(12r) einai h Crank Nicolson
{
     int i,j,t=0, j0=(int)(STEP/k) , j1=6*j0+1;
     double a[n],b[n],c[n],d[n], v1,v2,v=1-2*r*(1+h);

     //Arxikes sin8ikes
     for (i=0;i<=n;i++) U[i][0]=1;

     for (j=1;j<=m;j++)
     {   printf("[%d]\r",j);
         //Sinoriakes Sin8ikes
         U[0][1]= v*U[0][0]+2*r*U[1][0];
         U[n][1]= v*U[n][0]+2*r*U[n-1][0];
         for (i=1;i<n;i++)
         {
             v1=fun_a(i+0.5,h,fa); v2=fun_a(i-0.5,h,fa);
             a[i]=-l*r*v2;
             b[i]=1+l*r*(v1+v2);
             c[i]=-l*r*v1;
             d[i]=(1-l)*r*v2*U[i-1][0] + (1-(1-l)*r*(v1+v2))*U[i][0]
             + (1-l)*r*v1*U[i+1][0];
         }
         d[1]  -=a[1]*U[0][1];
         d[n-1]-=c[n-1]*U[n][1];

     thomas(n,U,a,b,c,d);
     for (i=0;i<=n;i++) U[i][0]=U[i][1];
     if ( (j%j0==0) && (j<j1) ) sample(U,n,j,j0);
     }
}
```

# 3 Πειραματικά αποτελέσματα

Με τη ρουτίνα `test.c` (η οποία έχει ως είσοδο το $h$) λαμβάνονται δειγματοληπτικά αποτελέσματα από τις παραπάνω μεθόδους για για $r = 0.1, 0.25, 0.48, 0.5, 0.52, 0.8, 1.0, 2.0$. Η δειγματοληψία γίνεται με τη συνάρτηση `sample.c` στους χρόνους $t = STEP, 2*STEP, \ldots, 6*STEP$ (όπου η σταθερά $STEP$ καθορίζεται στο πρόγραμμα) και σε έξι ισαπέχοντα σημεία πάνω στον $x-$άξονα. Γίνονται όλοι οι συνδυασμοί για τις διαφορετικές $a(x)$ και για τις δυο διαφορετικές αρχικές συνθήκες. Τα πειραματικά αποτελέσματα αποθηκεύονται στο αρχείο test.txt.

Έγιναν οι εξής εκτελέσεις:

- test10.txt Για $h = \frac{1}{10}$, δειγματοληπτικά $t = 0.05, 0.10, \ldots, 0.30$.
- test100.txt Για $h = \frac{1}{100}$, δειγματοληπτικά $t = 0.05, 0.10, \ldots, 0.30$
- test1000.txt Για $h = \frac{1}{1000}$, δειγματοληπτικά $t = 0.025, 0.05, \ldots, 0.15$.

Ενδεικτικά παραθέτουμε μερικούς δειγματοληπτικούς πίνακες που πήραμε για $a(x) = 1$, $h = \frac{1}{1000}$ και $r = 0.5,\ 0.52$. Στην πρώτη γραμμή φαίνεται το χωρικό βήμα $i$ από το οποίο προέρχεται η τιμή και στην πρώτη στήλη ο χρόνος $t$ και το χρονοβήμα $j$.

| $t$ | $j$ | $i = 166$ | $i = 332$ | $i = 498$ | $i = 664$ | $i = 830$ | $i = 996$ |
|---|---|---|---|---|---|---|---|
| **Άμεση μέθοδος**($r = 0.5$) | | | | | | | |
| 0.05 | 50000 | 0.26943958 | 0.43240632 | 0.48691997 | 0.47606084 | 0.35509429 | 0.01059302 |
| 0.10 | 100000 | 0.19606208 | 0.34771839 | 0.42525066 | 0.40989105 | 0.27485671 | 0.00762846 |
| 0.15 | 150000 | 0.15693328 | 0.28536919 | 0.35694321 | 0.34430053 | 0.22601585 | 0.00617814 |
| 0.20 | 200000 | 0.12883139 | 0.23590204 | 0.29712751 | 0.28729073 | 0.18805447 | 0.00512685 |
| 0.25 | 250000 | 0.10654760 | 0.19554342 | 0.24694690 | 0.23920164 | 0.15664816 | 0.00427013 |
| 0.30 | 300000 | 0.08834717 | 0.16228976 | 0.20520077 | 0.19898834 | 0.13040795 | 0.00355571 |
| **Crank-Nicolson**($r = 0.5$) | | | | | | | |
| 0.05 | 50000 | 0.26943982 | 0.43240562 | 0.48691890 | 0.47605989 | 0.35509424 | 0.01059305 |
| 0.10 | 100000 | 0.19606216 | 0.34771811 | 0.42524991 | 0.40989042 | 0.27485665 | 0.00762846 |
| 0.15 | 150000 | 0.15693326 | 0.28536901 | 0.35694284 | 0.34430018 | 0.22601573 | 0.00617814 |
| 0.20 | 200000 | 0.12883137 | 0.23590195 | 0.29712735 | 0.28729057 | 0.18805438 | 0.00512685 |
| 0.25 | 250000 | 0.10654760 | 0.19554341 | 0.24694686 | 0.23920160 | 0.15664813 | 0.00427013 |
| 0.30 | 300000 | 0.08834719 | 0.16228980 | 0.20520082 | 0.19898837 | 0.13040797 | 0.00355571 |
| **Douglas**($r = 0.5$) | | | | | | | |
| 0.05 | 50000 | 0.26943974 | 0.43240585 | 0.48691926 | 0.47606021 | 0.35509426 | 0.01059304 |
| 0.10 | 100000 | 0.19606213 | 0.34771820 | 0.42525016 | 0.40989063 | 0.27485667 | 0.00762846 |
| 0.15 | 150000 | 0.15693327 | 0.28536907 | 0.35694296 | 0.34430030 | 0.22601577 | 0.00617814 |
| 0.20 | 200000 | 0.12883138 | 0.23590198 | 0.29712741 | 0.28729062 | 0.18805441 | 0.00512685 |
| 0.25 | 250000 | 0.10654760 | 0.19554341 | 0.24694687 | 0.23920161 | 0.15664814 | 0.00427013 |
| 0.30 | 300000 | 0.08834718 | 0.16228978 | 0.20520080 | 0.19898836 | 0.13040796 | 0.00355571 |
| **Άμεση μέθοδος(συνθήκες με παραγώγους)**($r = 0.5$) | | | | | | | |
| 0.05 | 50000 | 0.94608867 | 0.98785552 | 0.99825480 | 0.99730748 | 0.97810294 | 0.89133040 |
| 0.10 | 100000 | 0.89554001 | 0.95740703 | 0.98400743 | 0.98323071 | 0.94848856 | 0.85204070 |
| 0.15 | 150000 | 0.85576899 | 0.92515742 | 0.96043550 | 0.96178386 | 0.92201671 | 0.82374761 |
| 0.20 | 200000 | 0.82187738 | 0.89373650 | 0.93326239 | 0.93697990 | 0.89691115 | 0.79981634 |
| 0.25 | 250000 | 0.79153561 | 0.86362075 | 0.90513414 | 0.91090142 | 0.87232043 | 0.77762586 |
| 0.30 | 300000 | 0.76359422 | 0.83485896 | 0.87716374 | 0.88454924 | 0.84798653 | 0.75613155 |
| **Crank-Nicolson(συνθήκες με παραγώγους)**($r = 0.5$) | | | | | | | |
| 0.05 | 50000 | 0.94608827 | 0.98785510 | 0.99825457 | 0.99730727 | 0.97810265 | 0.89133021 |
| 0.10 | 100000 | 0.89553977 | 0.95740669 | 0.98400704 | 0.98323038 | 0.94848835 | 0.85204058 |
| 0.15 | 150000 | 0.85576880 | 0.92515713 | 0.96043514 | 0.96178353 | 0.92201650 | 0.82374748 |
| 0.20 | 200000 | 0.82187720 | 0.89373624 | 0.93326208 | 0.93697959 | 0.89691093 | 0.79981618 |
| 0.25 | 250000 | 0.79153544 | 0.86362052 | 0.90513386 | 0.91090114 | 0.87232020 | 0.77762568 |
| 0.30 | 300000 | 0.76359406 | 0.83485876 | 0.87716350 | 0.88454899 | 0.84798630 | 0.75613136 |
| **Douglas(συνθήκες με παραγώγους)**($r = 0.5$) | | | | | | | |
| 0.05 | 50000 | 0.94608840 | 0.98785524 | 0.99825464 | 0.99730734 | 0.97810275 | 0.89133027 |
| 0.10 | 100000 | 0.89553985 | 0.95740680 | 0.98400717 | 0.98323049 | 0.94848842 | 0.85204062 |
| 0.15 | 150000 | 0.85576886 | 0.92515723 | 0.96043526 | 0.96178364 | 0.92201657 | 0.82374752 |
| 0.20 | 200000 | 0.82187726 | 0.89373633 | 0.93326218 | 0.93697970 | 0.89691100 | 0.79981623 |
| 0.25 | 250000 | 0.79153550 | 0.86362060 | 0.90513395 | 0.91090123 | 0.87232028 | 0.77762574 |
| 0.30 | 300000 | 0.76359412 | 0.83485883 | 0.87716358 | 0.88454907 | 0.84798638 | 0.75613142 |

| $t$ | $j$ | $i = 166$ | $i = 332$ | $i = 498$ | $i = 664$ | $i = 830$ | $i = 996$ |
|---|---|---|---|---|---|---|---|
| **Crank-Nicolson**($r = 0.52$) | | | | | | | |
| 0.05 | 48076 | 0.26944207 | 0.43240759 | 0.48691975 | 0.47606101 | 0.35509651 | 0.01059314 |
| 0.10 | 96152 | 0.19606406 | 0.34772087 | 0.42525263 | 0.40989309 | 0.27485891 | 0.00762853 |
| 0.15 | 144228 | 0.15693512 | 0.28537220 | 0.35694659 | 0.34430376 | 0.22601817 | 0.00617821 |
| 0.20 | 192304 | 0.12883328 | 0.23590539 | 0.29713159 | 0.28729461 | 0.18805703 | 0.00512692 |
| 0.25 | 240380 | 0.10654954 | 0.19554693 | 0.24695127 | 0.23920583 | 0.15665090 | 0.00427020 |
| 0.30 | 288456 | 0.08834910 | 0.16229329 | 0.20520521 | 0.19899261 | 0.13041074 | 0.00355579 |
| **Douglas**($r = 0.52$) | | | | | | | |
| 0.05 | 48076 | 0.26944199 | 0.43240782 | 0.48692010 | 0.47606133 | 0.35509653 | 0.01059314 |
| 0.10 | 96152 | 0.19606404 | 0.34772096 | 0.42525288 | 0.40989329 | 0.27485893 | 0.00762853 |
| 0.15 | 144228 | 0.15693513 | 0.28537226 | 0.35694671 | 0.34430387 | 0.22601821 | 0.00617821 |
| 0.20 | 192304 | 0.12883329 | 0.23590541 | 0.29713164 | 0.28729467 | 0.18805706 | 0.00512692 |
| 0.25 | 240380 | 0.10654954 | 0.19554693 | 0.24695128 | 0.23920585 | 0.15665091 | 0.00427020 |
| 0.30 | 288456 | 0.08834909 | 0.16229327 | 0.20520520 | 0.19899260 | 0.13041073 | 0.00355579 |
| **Crank-Nicolson(συνθήκες με παραγώγους)**($r = 0.52$) | | | | | | | |
| 0.05 | 48076 | 0.94608937 | 0.98785560 | 0.99825470 | 0.99730743 | 0.97810324 | 0.89133114 |
| 0.10 | 96152 | 0.89554145 | 0.95740791 | 0.98400781 | 0.98323109 | 0.94848940 | 0.85204180 |
| 0.15 | 144228 | 0.85577087 | 0.92515896 | 0.96043663 | 0.96178488 | 0.92201796 | 0.82374893 |
| 0.20 | 192304 | 0.82187963 | 0.89373859 | 0.93326421 | 0.93698156 | 0.89691282 | 0.79981792 |
| 0.25 | 240380 | 0.79153821 | 0.86362334 | 0.90513656 | 0.91090366 | 0.87232254 | 0.77762775 |
| 0.30 | 288456 | 0.76359716 | 0.83486198 | 0.87716668 | 0.88455201 | 0.84798908 | 0.75613380 |
| **Douglas(συνθήκες με παραγώγους)**($r = 0.52$) | | | | | | | |
| 0.05 | 48076 | 0.94608950 | 0.98785574 | 0.99825477 | 0.99730750 | 0.97810334 | 0.89133121 |
| 0.10 | 96152 | 0.89554152 | 0.95740802 | 0.98400794 | 0.98323120 | 0.94848948 | 0.85204184 |
| 0.15 | 144228 | 0.85577094 | 0.92515906 | 0.96043675 | 0.96178499 | 0.92201803 | 0.82374898 |
| 0.20 | 192304 | 0.82187969 | 0.89373868 | 0.93326432 | 0.93698166 | 0.89691289 | 0.79981797 |
| 0.25 | 240380 | 0.79153827 | 0.86362341 | 0.90513665 | 0.91090375 | 0.87232262 | 0.77762782 |
| 0.30 | 288456 | 0.76359721 | 0.83486205 | 0.87716676 | 0.88455209 | 0.84798916 | 0.75613387 |

**Άμεση μέθοδος**($r = 0.52$): ο πίνακας είναι γεμάτος $-1.\#IND000$.

**Άμεση μέθοδος(συνθήκες με παραγώγους)**($r = 0.52$): ο πίνακας είναι γεμάτος $-1.\#IND000$.

Τα αποτελέσματα αναδεικνύουν ότι η άμεση μέθοδος δίνει σωστά αποτελέσματα μόνο για $r \leq \frac{1}{2}$. Στην δεύτερη εκτέλεση όπου δεν ικανοποιείται αυτή η απαίτηση παίρνουμε τεράστιες τιμές, υπερχείλιση ή υπεκχείλιση υπολογισμών.

Οι μέθοδοι Crank-Nicolson και Douglas δίνουν εφάμιλλα αποτελέσματα τις περισσότερες φορές. Παρατηρείται οι τιμές της μεθόδου Douglas να βρίσκονται συνήθως ενδιάμεσα σε αυτές της άμεσης μεθόδου και της μεθόδου Crank-Nicolson.

Οι μέθοδοι έχουν καλύτερα αποτελέσματα για σταθερές συνοριακές συνθήκες. Για συνθήκες με παραγώγους, προβλήματα ευστάθειας παρουσιάζονται ακόμη και στις έμμεσες μεθόδους, για $r \geq 1$(βλέπε και ενότητα 4).

Αν μικρύνουμε αρκετά το βήμα $h$, πχ $h = \frac{1}{10000}$, βλέπουμε ότι για να διατηρήσουμε το $r$ σε μικρές τιμές πρέπει να λάβουμε πολύ μικρό χρονοβήμα $k$. Για παράδειγμα:

**Douglas**($r=\frac{1}{2},\ h=\frac{1}{10000}$)

| $t$ | $j\backslash i$ | 1666 | 3332 | 4998 | 6664 | 8330 | 9996 |
|---|---|---|---|---|---|---|---|
| 0.0002 | 40000 | 0.50000000 | 0.50000000 | 0.50000000 | 0.50000000 | 0.50000000 | 0.00797836 |
| 0.0004 | 80000 | 0.50000000 | 0.50000000 | 0.50000000 | 0.50000000 | 0.50000000 | 0.00564173 |
| 0.0006 | 120000 | 0.49999924 | 0.50000000 | 0.50000000 | 0.50000000 | 0.49999929 | 0.00460650 |
| 0.0008 | 160000 | 0.49998443 | 0.50000000 | 0.50000000 | 0.50000000 | 0.49998510 | 0.00398936 |
| 0.0010 | 200000 | 0.49990246 | 0.50000000 | 0.50000000 | 0.50000000 | 0.49990586 | 0.00356821 |
| 0.0012 | 240000 | 0.49966395 | 0.50000000 | 0.50000000 | 0.50000000 | 0.49967385 | 0.00325732 |

χρειάστηκαν 240000 επαναλήψεις για να υπολογίσουμε τις τιμές σε χρόνο μόλις $t$=0.0012. Προφανώς τόσο μικρό χρονικό βήμα δεν έχει νόημα στις εφαρμογές, και πράγματι βλέπουμε πως ο παραπάνω πίνακας δεν έχει αξιόλογες πληροφορίες.

# 4 Πειραματική εύρεση πεδίου σύγκλισης της Crank-Nicolson

Η συνάρτηση `cn_conv.c` έχει είσοδο μια ακρίβεια $tol$. Αρχικά υπολογίζει έναν «ακριβή» πίνακα (δειγματοληπτικών) τιμών $D$ με τη μέθοδο Douglas. Κατόπιν τρέχει τη μέθοδο CN για διάφορες τιμές του $r$ (και για $h=0.01$,), ώστε να λάβουμε έναν αντίστοιχο πίνακα $C$. Η διαδικασία επαναλαμβάνεται για αυξανόμενα $r$ έως ότου συμβεί πέντε φορές να είναι

$$\|D-C\|_\infty > tol$$

Στην οθόνη τυπώνονται τα $r$ και το αντίστοιχο σφάλμα $\|D-C\|_\infty$. Για σταθερές συνοριακές συνθήκες και $a(x)=1$ πήραμε:

| $tol=10^{-3}$ | | $tol=10^{-2}$ | | $tol=10^{-1}$ | |
|---|---|---|---|---|---|
| $r$ | $\|D-C\|_\infty$ | $r$ | $\|D-C\|_\infty$ | $r$ | $\|D-C\|_\infty$ |
| 0.30 | 0.00029786 | 5.10 | 0.00400564 | 45.20 | 0.03871160 |
| 0.60 | 0.00052758 | 5.20 | 0.00500946 | 45.30 | 0.03705417 |
| 1.00 | 0.00052762 | 5.40 | 0.00887706 | 45.40 | 0.03539877 |
| 1.10 | 0.00152344 | 5.50 | 0.01174610 | 45.50 | 0.11060484 |
| 1.20 | 0.00190665 | 5.70 | 0.01050488 | 45.70 | 0.10658718 |
| 1.30 | 0.00198331 | 5.90 | 0.01112557 | 45.90 | 0.10266769 |

Παρατηρώντας τα αποτελέσματα βλέπουμε ότι:
- Για $tol=10^{-3}$ παίρνουμε ικανοποιητικές τιμές περίπου για $r\leq 1$.
- Για $tol=10^{-2}$ παίρνουμε ικανοποιητικές τιμές περίπου για $r\leq 5$.
- Για $tol=10^{-1}$ παίρνουμε ικανοποιητικές τιμές περίπου για $r\leq 50$.

Συμπερασματικά η Crank-Nicolson μπορεί να θεωρηθεί αρκετά ακριβής για $r\leq 5$.

Τα πράγματα αλλάζουν όταν έχουμε συνοριακές συνθήκες με παραγώγους($a(x)=1$):

| $tol=10^{-1}$ | |
|---|---|
| $r$ | $\|D-C\|_\infty$ |
| 0.50 | 0.00049244 |
| 0.70 | 0.00019094 |
| 0.80 | 0.00058430 |
| 0.90 | 0.00122685 |
| 1.00 | 171271.1129 |
| 1.10 | 5587420... |

Εδώ η μέθοδος είναι ακριβής μόνο για $r<1$.