Numerical Methods for Financial Derivatives

Hwan C. Lin
Department of Economics
UNC Charlotte

Lecture 12: Iterative Methods for Linear Systems of Equations

Iterative Solvers:

- Jacobi Method
- Gauss-Seidel Method
- SOR (Successive Over Relaxation)
    - Gauss-Seidel is an extension of Jacobi
    - SOR is an extension of Gauss-Seidel
    - The SOR method plays a pivotal role in pricing American-style options.

Fixed-Point Iteration: One Equation

- General Form:

$$f(x) = 0$$

$$\text{e.g.,} \quad f(x) = x^2 - 2x - 3 = 0$$

- If $x^*$is a root of $f(x) = 0$, then

$$f(x^*) = 0 \Rightarrow x^* = g(x^*)$$

  where x^* is called a fixed point for the function $g(x)$.

- Use of fixed-point iteration to find x^*:
  - Case 1:

$$x = g_1(x) = \sqrt{2x+3}$$

  - Case 2:

$$x = g_2(x) = \frac{3}{x-2}$$

  - Case 3:

$$x = g_3(x) = \frac{x^2-3}{2}$$

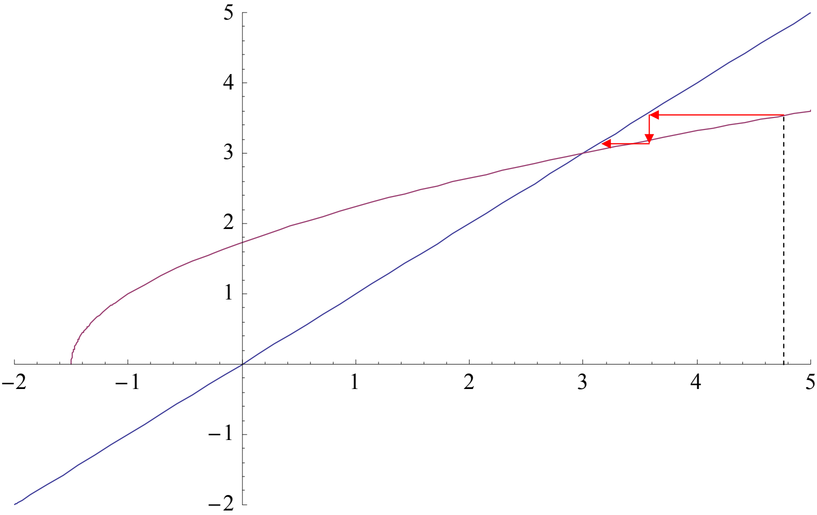
- Case 1:

$$x = g_1(x) = \sqrt{2x+3}$$

$$x^{(k+1)} = g_1(x^{(k)}) = \sqrt{2x^{(k)}+3}$$

- Iteration:

$$\begin{aligned} x^{(0)} &= 4 \\ x^{(1)} &= 3.31662 \\ x^{(2)} &= 3.10375 \\ x^{(3)} &= 3.03439 \\ x^{(4)} &= 3.01144 \\ x^{(5)} &= 3.00381 \\ x^{(6)} &= 3.00127 \end{aligned}$$


5
4
3
2
1
−1
−2
−2
−1
1
2
3
4
5

- Case 2:

$$x = g_2(x) = \frac{3}{x-2}$$

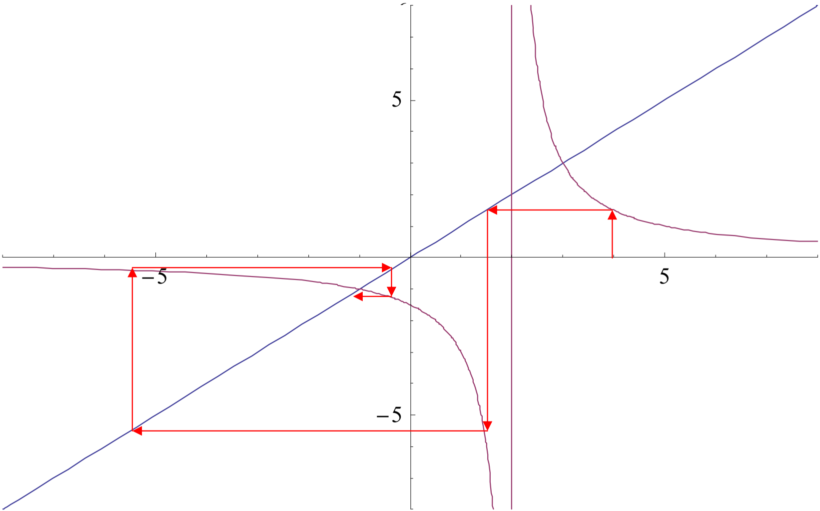
$$x^{(k+1)} = g_2(x^{(k)}) = \frac{3}{x^{(n)} - 2}$$

- Iteration:

$$x^{(0)} = 4$$
$$x^{(1)} = 1.5$$
$$x^{(2)} = -6$$
$$x^{(3)} = -0.375$$
$$x^{(4)} = -1.26316$$
$$x^{(5)} = -0.919355$$
$$x^{(6)} = -1.02762$$
$$x^{(7)} = -0.990876$$
$$x^{(8)} = -1.00305$$

Fixed-Point Iteration: One Equation

Case 2: Oscillatory Convergence (2)



- Case 3:

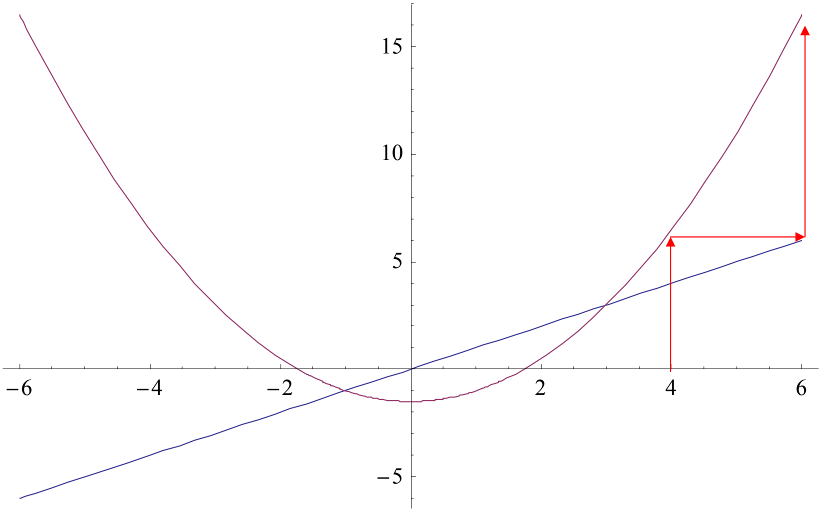
$$x = g_3(x) = \frac{x^2 - 3}{2}$$

$$x^{(k+1)} = g_3(x^{(k)}) = \frac{(x^{(k)})^2 - 3}{2}$$

- Iteration:

$$x^{(0)} = 4$$
$$x^{(1)} = 6.5$$
$$x^{(2)} = 19.625$$
$$x^{(3)} = 191.07$$
$$x^{(4)} = 18,252.04$$
$$x^{(5)} = 1.66576 * 10^8$$

Fixed-Point Iteration: One Equation
Case 3: Divergence (2)



Solution

[Iteration with the form $x = g(x)$] To determine a root of $f(x)$=0, given a value, reasonably close to the root, rearrange the equation to an equivalent form x= g(x).

REPEAT

Set $x^{(2)} = x^{(1)}$.

Set $x^{(1)} = g(x^{(1)})$.

UNTIL $|x^{(1)} - x^{(2)}| <$ tolerance value.

Lemma

If $g(x)$ and $g'(x)$ are continuous on an interval about the fixed point x^* of $x = g(x)$, and if $|g'(x)| < 1$ for all x in the interval, then $x^{(k+1)} = g(x^{(k)})$, $k = 0, 1, 2, \ldots$, will converge to x^*, provided that is chosen in the interval.

Fixed-Point Iteration: Set of Linear Equations

- Linear System: $A \cdot x = b$

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix}$$

- Partition of A into D, L, U so that

$$A = D - L - U$$

- The Diagonal Matrix, Strictly Lower Triangular, and Strictly Upper Triangular:

$$A = \underbrace{\begin{bmatrix} a_{\mathbf{11}} & & & 0 \\ & a_{\mathbf{22}} & & \\ & & \ddots & \\ 0 & & & a_{NN} \end{bmatrix}}_{D} - \underbrace{\begin{bmatrix} 0 & \cdots & \cdots & 0 \\ -a_{\mathbf{21}} & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ -a_{N\mathbf{1}} & \cdots & -a_{N,N-\mathbf{1}} & 0 \end{bmatrix}}_{L} - \underbrace{\begin{bmatrix} 0 & -a_{\mathbf{12}} & \cdots & -a_{\mathbf{1}N} \\ \vdots & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & -a_{N-\mathbf{1},N} \\ 0 & \cdots & \cdots & 0 \end{bmatrix}}_{U}$$

Fixed-Point Iteration: Set of Linear Equations (2)

- **Jacobi Method**

$$\text{Iteration Eqs} \quad : \quad x_i^{(k+1)} = \frac{1}{a_{ii}}\Big(b_i - \sum_{\substack{j=1 \\ j\neq i}}^{N} a_{ij}x_j^{(k)}\Big),$$

$$\text{Matrix Form} \quad : \quad x^{(k+1)} = D^{-1}[b + (L+U)\cdot x^{(k)}], \ \ i = 1,2,\dots,N$$

- **Gauss-Seidel Method**

$$\text{Iteration Eqs} \quad : \quad x_i^{(k+1)} = \frac{1}{a_{ii}}\Big(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{N} a_{ij}x_j^{(k)}\Big),$$

$$\text{Matrix Form} \quad : \quad x^{(k+1)} = (D-L)^{-1}[b + U\cdot x^{(k)}]$$

- **SOR Method**

$$\begin{aligned}\text{Iteration Eqs} \quad : \quad x_i^{(k+1)} &= (1-\omega)x_i^{(k)} + \frac{\omega}{a_{ii}}\Big(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{N} a_{ij}x_j^{(k)}\Big) \\ &= x_i^{(k)} + \frac{\omega}{a_{ii}}\Big(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i}^{N} a_{ij}x_j^{(k)}\Big),\end{aligned}$$

$$\text{Matrix Form} \quad : \quad x^{(k+1)} = (D-\omega L)^{-1}[\omega b + \omega U\cdot x^{(k)} + (1-\omega)D\cdot x^{(k)}]$$

- **Jacobi Method:**

$$D \cdot x^{(k+1)} = b + (L+U) \cdot x^{(k)}$$

$$\Rightarrow x^{(k+1)} = D^{-1}[b + (L+U)] \cdot x^{(k)}$$

- **Gauss-Seidel Method:**

$$D \cdot x^{(k+1)} = b + L \cdot x^{(k+1)} + U \cdot x^{(k)}$$

$$\Rightarrow (D-L) \cdot x^{(k+1)} = b + U \cdot x^{(k)}$$

$$\Rightarrow x^{(k+1)} = (D-L)^{-1}[b + U \cdot x^{(k)}]$$

- **SOR Method:** **(**ω is the relaxation parameter)

$$D \cdot x^{(k+1)} = (1-\omega) D \cdot x^{(k)} + \omega[b + L \cdot x^{(k+1)} + U \cdot x^{(k)}]$$

$$(D - \omega L) \cdot x^{(k+1)} = \omega b + \omega U \cdot x^{(k)} + (1-\omega) D \cdot x^{(k)}$$

$$x^{(k+1)} = (D - \omega L)^{-1}[\omega b + \omega U \cdot x^{(k)} + (1-\omega) D \cdot x^{(k)}]$$

- **Iteration Equations:**

$$x^{(k+1)} = G \cdot x^{(k)} + c$$

- **Iteration Matrix**

$$\begin{aligned} \textbf{Jacobi:} &\quad G = D^{-1}(L+U) \\ \textbf{Gauss-Seidel:} &\quad G = (D-L)^{-1}U \\ \textbf{SOR:} &\quad G = (D-\omega L)^{-1}[\omega U + (1-\omega)D] \end{aligned}$$

- **Lemma:** The iteration $x^{(k+1)} = G \cdot x^{(k)} + c$ converges as $k \to \infty$ for all starting vectors x iff

$$\rho(G) < 1,$$

where $\rho(G)$ is the spectral radius of G. The asymptotic convergence rate is $-\ln \rho$. (e.g., to reduce the error by a factor of 10^p, the number of iterations required will be approximately $\ln 10^p / (-\ln \rho)$).

- **Proof:** The error after n iterations

$$e^{(k+1)} = G \cdot e^{(k)} = G^2 \cdot e^{(k-1)} = \ldots = G^{k+1} \cdot e^{(0)}$$

- Given

$$A \cdot x = b,$$

$$x^{(k+1)} = G \cdot x^{(k)} + c,$$

  Both Jacobi and Gauss-Seidel converges,if matrix A is strictly diagonally dominant for all rows ($|a_{ii}| > \sum_{i \neq j} |a_{ij}|, \forall i$),

- The requirement of diagonal dominance is only a sufficient condition but not a necessary condition. Without diagonal dominance, neither Jacobi nor Gauss-Seidel is sure to converge.
- When both Jacobi and Gauss-Seidel converge, the latter converges faster than Jacobi.
- When matrix A is symmetric and positive definite, Gauss-Seidel will converge from any starting vector.
- SOR does not converge unless $0 < \omega < 2$.

Example: Jacobi

- **Linear System**: $A \cdot x = b$

$$\begin{aligned} 6x_1 - 2x_2 + x_3 &= 11 \\ x_1 + 2x_2 - 5x_3 &= -1 \\ -2x_1 + 7x_2 + 2x_3 &= 5 \end{aligned} \quad \Rightarrow \quad \begin{aligned} \mathbf{6x}_1 - 2x_2 + x_3 &= 11 \\ -2x_1 + \mathbf{7x}_2 + 2x_3 &= 5 \\ x_1 + 2x_2 - \mathbf{5x}_3 &= -1 \end{aligned}$$

  where $A \cdot x = b$ is re-ordered to be diagonally dominant.

- **Iteration Equations**:

$$x_i^{(k+1)} = \frac{1}{a_{ii}}(b_i - \sum_{\substack{j=1 \\ j \neq i}}^{N} a_{ij} x_j^{(k)})$$

$$\begin{aligned} x_1 &= \tfrac{11}{6} - \tfrac{1}{6}(-2x_2 + x_3) \\ x_2 &= \tfrac{5}{7} - \tfrac{1}{7}(-2x_1 + 2x_3) \\ x_3 &= \tfrac{1}{5} - \tfrac{1}{5}(-x_1 - 2x_2) \end{aligned} \quad \Rightarrow \quad \begin{aligned} x_1^{(k+1)} &= 1.8333 + 0.3333x_2^{(k)} - 0.1677x_3^{(k)} \\ x_2^{(k+1)} &= 0.7143 + 0.2857x_1^{(k)} - 0.2857x_3^{(k)} \\ x_3^{(k+1)} &= 0.2000 + 0.2000x_1^{(k)} + 0.4000x_2^{(k)} \end{aligned}$$

Example: Jacobi (2)

- Matrix Form, $x^{(k+1)} = c + G \cdot x^{(k)}$, where $c = D^{-1}b, G = D^{-1}(L+U)$:

$$\underbrace{\begin{bmatrix} x_1^{(k+1)} \\ x_2^{(k+1)} \\ x_3^{(k+1)} \end{bmatrix}}_{x^{(k+1)}} = \underbrace{\begin{bmatrix} \frac{11}{6} \\ \frac{5}{7} \\ \frac{1}{5} \end{bmatrix}}_{c} + \underbrace{\begin{bmatrix} 0 & \frac{2}{6} & -\frac{1}{6} \\ \frac{2}{7} & 0 & -\frac{2}{7} \\ \frac{1}{5} & \frac{2}{5} & 0 \end{bmatrix}}_{G} \underbrace{\begin{bmatrix} x_1^{(k)} \\ x_2^{(k)} \\ x_3^{(k)} \end{bmatrix}}_{x^{(k)}}$$

- Results of Iterations:

	First	Second	Third	Fourth	Fifth	Sixth	$\cdots$	Ninth
$\mathbf{x}_1$	0	1.833	2.038	2.085	2.004	1.994	$\cdots$	2.000
$\mathbf{x}_2$	0	0.714	1.181	1.053	1.001	0.990	$\cdots$	1.000
$\mathbf{x}_3$	0	0.200	0.852	1.080	1.038	1.001	$\cdots$	1.000

- Partition A into $A = D - L - U$:

$$\underbrace{\begin{bmatrix} 6 & -2 & 1 \\ -2 & 7 & 2 \\ 1 & 2 & -5 \end{bmatrix}}_{A} = \underbrace{\begin{bmatrix} 6 & 0 & 0 \\ 0 & 7 & 0 \\ 0 & 0 & -5 \end{bmatrix}}_{D} - \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 0 \\ -1 & -2 & 0 \end{bmatrix}}_{L} - \underbrace{\begin{bmatrix} 0 & 2 & -1 \\ 0 & 0 & -2 \\ 0 & 0 & 0 \end{bmatrix}}_{U}$$

- Check $c = D^{-1}b$:

$$c = D^{-1}b = \begin{bmatrix} 1/6 & 0 & 0 \\ 0 & 1/7 & 0 \\ 0 & 0 & -1/5 \end{bmatrix} \begin{bmatrix} 11 \\ 5 \\ -1 \end{bmatrix} = \begin{bmatrix} 11/6 \\ 5/7 \\ 1/5 \end{bmatrix}$$

- Check $G = D^{-1}(L + U)$:

$$G = \begin{bmatrix} 1/6 & 0 & 0 \\ 0 & 1/7 & 0 \\ 0 & 0 & -1/5 \end{bmatrix} \begin{bmatrix} 0 & 2 & -1 \\ 2 & 0 & -2 \\ -1 & -2 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 2/6 & -1/6 \\ 2/7 & 0 & -2/7 \\ 1/5 & 2/5 & 0 \end{bmatrix}$$

Example: Gauss-Seidel

- Iteration Equations:

$$
\begin{aligned}
x_i^{(k+1)} &= \frac{1}{a_{ii}}(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{N} a_{ij} x_j^{(k)}), \text{ or} \\
x^{(k+1)} &= (D-L)^{-1}[b + U \cdot x^{(k)}]
\end{aligned}
$$

- Example:

$$
\begin{aligned}
x_1^{(k+1)} &= 1.8333 + 0.3333 x_2^{(k)} - 0.1677 x_3^{(k)} \\
x_2^{(k+1)} &= 0.7143 + 0.2857 x_1^{(k+1)} - 0.2857 x_3^{(k)} \\
x_3^{(k+1)} &= 0.2000 + 0.2000 x_1^{(k+1)} + 0.4000 x_2^{(k+1)}
\end{aligned}
$$

$$\Downarrow$$

$$
\begin{aligned}
x_1^{(k+1)} &= 1.8333 + 0.3333 x_2^{(k)} - 0.1677 x_3^{(k)} \\
x_2^{(k+1)} &= 1.2381 + 0.0952 x_2^{(k)} - 0.3333 x_3^{(k)} \\
x_3^{(k+1)} &= 1.0619 + 0.1048 x_2^{(k)} - 0.1667 x_3^{(k)}
\end{aligned}
$$

Example: Gauss-Seidel (2)

- Results:

	First	Second	Third	Fourth	Fifth	$\cdots$	Sixth
$\mathbf{x}_1$	0	1.833	2.069	1.998	1.999	$\cdots$	2.000
$\mathbf{x}_2$	0	1.238	1.002	0.995	1.000	$\cdots$	1.000
$\mathbf{x}_3$	0	1.062	1.015	0.998	1.000	$\cdots$	1.000

Example: Gauss-Seidel (3)

- Check $c = (D - L)^{-1} b$:

$$c = \begin{bmatrix} 0.1667 & 0 & 0 \\ 0.0476 & 0.1429 & 0 \\ 0.0524 & 0.0571 & -0.2 \end{bmatrix} \begin{bmatrix} 11 \\ 5 \\ -1 \end{bmatrix} = \begin{bmatrix} 1.8333 \\ 1.2381 \\ 1.0619 \end{bmatrix}$$

- Check $G = (D - L)^{-1} U$:

$$\begin{aligned} G &= \begin{bmatrix} 0.1667 & 0 & 0 \\ 0.0476 & 0.1429 & 0 \\ 0.0524 & 0.0571 & -0.2 \end{bmatrix} \begin{bmatrix} 0 & -2 & 1 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0.3333 & -0.1667 \\ 0 & 0.0952 & -0.3333 \\ 0 & 0.1048 & -0.1667 \end{bmatrix} \end{aligned}$$

Example: SOR

- A 4×4 linear system:

$$\begin{bmatrix} -4 & 1 & 1 & 1 \\ 1 & -4 & 1 & 1 \\ 1 & 1 & -4 & 1 \\ 1 & 1 & 1 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

- SOR: $x_i^{(k+1)} = x_i^{(k)} + \frac{\omega}{a_{ii}}[b_1 - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i}^{4} a_{ij} x_j^{(k)}]$

$$x_1^{(k+1)} = x_1^{(k)} + \frac{\omega}{a_{11}}[b_1 - a_{11}x_1^{(k)} - a_{12}x_2^{(k)} - a_{13}x_3^{(k)} - a_{14}x_4^{(k)}]$$

$$x_2^{(k+1)} = x_2^{(k)} + \frac{\omega}{a_{22}}[b_2 - a_{21}x_1^{(k+1)} - (a_{22}x_2^{(k)} + a_{23}x_3^{(k)} + a_{24}x_4^{(k)})]$$

$$x_3^{(k+1)} = x_3^{(k)} + \frac{\omega}{a_{33}}[b_3 - (a_{31}x_1^{(k+1)} + a_{32}x_2^{(k+1)}) - (a_{33}x_3^{(k)} + a_{34}x_4^{(k)})]$$

$$x_4^{(k+1)} = x_4^{(k)} + \frac{\omega}{a_{44}}[b_4 - (a_{41}x_1^{(k+1)} + a_{42}x_2^{(k+1)} + a_{43}x_3^{(k+1)}) - a_{44}x_4^{(k)})]$$

- Mathematica Program: Overrelaxation, Underrelaxation

Applying Iterative Methods to Tridiagonal Matrix

- Tridiagonal Matrix

$$\begin{bmatrix} \alpha_1 & \beta_1 & & & 0 \\ \gamma_2 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \gamma_{N-1} & \alpha_{N-1} & \beta_{N-1} \\ 0 & & & \gamma_N & \alpha_N \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_3 \\ b_4 \end{bmatrix}$$

- Python code: $\alpha = 2$, $\gamma = -1$, $\beta = 1$

```
import numpy as np
from scipy.sparse import spdiags
maindiag = 2 * np.ones(5)
subdiag  = - np.ones(5)
superdiag = np.ones(5)
data = np.array([maindiag, subdiag, superdiag])
triset = np.array([0, -1, 1])
A = spdiags(data, triset, 5, 5).toarray()
print "Matrix A is: "
print A
```