# Engineering Analysis ENG 3420 Fall 2009

Dan C. Marinescu
Office: HEC 439 B
Office hours: Tu-Th 11:00-12:00

# Lecture 20

- Last time:
  - The inverse of a matrix
  - Iterative methods for solving sytems of linear equations
    - Gauss-Siedel
    - Jacobi
- Today
  - Relaxation
  - Non-linear systems
  - Random variables, probability distributions, Matlab support for random variables.
- Next Time
  - Linear regression
  - Linear least squares regression

# Relaxation

- To enhance convergence, an iterative program can introduce *relaxation* where the value at a particular iteration is made up of a combination of the old value and the newly calculated value:

$$x_i^{\text{new}} = \lambda x_i^{\text{new}} + (1-\lambda) x_i^{\text{old}}$$

  where $\lambda$ is a weighting factor that is assigned a value between 0 and 2.
  - $0<\lambda<1$: underrelaxation
  - $\lambda=1$: no relaxation
  - $1<\lambda\leq 2$: overrelaxation

# Nonlinear Systems

- Nonlinear systems can also be solved using the same strategy as the Gauss-Seidel method - solve each system for one of the unknowns and update each unknown using information from the previous iteration.
- This is called *successive substitution*.

# Newton-Raphson

- Nonlinear systems may also be solved using the Newton-Raphson method for multiple variables.
- For a two-variable system, the Taylor series approximation and resulting Newton-Raphson equations are:

$$f_{1,i+1} = f_{1,i} + \left(x_{1,i+1} - x_{1,i}\right)\frac{\partial f_{1,i}}{\partial x_1} + \left(x_{2,i+1} - x_{2,i}\right)\frac{\partial f_{1,i}}{\partial x_2}$$

$$x_{1,i+1} = x_{1,i} - \frac{f_{1,i}\frac{\partial f_{2,i}}{\partial x_2} - f_{2,i}\frac{\partial f_{1,i}}{\partial x_2}}{\frac{\partial f_{1,i}}{\partial x_1}\frac{\partial f_{2,i}}{\partial x_2} - \frac{\partial f_{1,i}}{\partial x_2}\frac{\partial f_{2,i}}{\partial x_1}}$$

$$f_{2,i+1} = f_{2,i} + \left(x_{1,i+1} - x_{1,i}\right)\frac{\partial f_{2,i}}{\partial x_1} + \left(x_{2,i+1} - x_{2,i}\right)\frac{\partial f_{2,i}}{\partial x_2}$$

$$x_{2,i+1} = x_{2,i} - \frac{f_{2,i}\frac{\partial f_{1,i}}{\partial x_1} - f_{1,i}\frac{\partial f_{2,i}}{\partial x_1}}{\frac{\partial f_{1,i}}{\partial x_1}\frac{\partial f_{2,i}}{\partial x_2} - \frac{\partial f_{1,i}}{\partial x_2}\frac{\partial f_{2,i}}{\partial x_1}}$$

```
function [x,f,ea,iter]=newtmult(func,x0,es,maxit,varargin)
% newtmult: Newton-Raphson root zeroes nonlinear systems
%   [x,f,ea,iter]=newtmult(func,x0,es,maxit,p1,p2,...):
%     uses the Newton-Raphson method to find the roots of
%     a system of nonlinear equations
% input:
%   func = function handle to function that returns f and J
%   x0 = initial guess
%   es = desired percent relative error (default = 0.0001%)
%   maxit = maximum allowable iterations (default = 50)
%   p1,p2,... = additional parameters used by function
% output:
%   x = vector of roots
%   f = vector of functions evaluated at roots
%   ea = approximate percent relative error (%)
%   iter = number of iterations

if nargin<2,error('at least 2 input arguments required'),end
if nargin<3||isempty(es),es=0.0001;end
if nargin<4||isempty(maxit),maxit=50;end
iter = 0;
x=x0;
while (1)
  [J,f]=func(x,varargin{:});
  dx=J\f;
  x=x-dx;
  iter = iter + 1;
  ea=100*max(abs(dx./x));
  if iter>=maxit||ea<=es, break, end
end
```

# Probability and statistics concepts

- See class notes:
  - Probability
  - NASA lecture