In [1]:
```
%matplotlib notebook
from numpy import *
from matplotlib.pyplot import *
```

Homework 10 : Conditioning and stability of linear least squares

The least squares problem

$$A\mathbf{x} = \mathbf{b} \tag{1}$$

where $A \in \mathcal{R}^{m \times n}$, $m \ge n$ has four associated "conditioning" problems, described in the table in Theorem 18.1 of TB (page 131). These are

1. Sensitivity of $\mathbf{y} = A\mathbf{x}$ to right hand side vector $\mathbf{b}$,
2. Sensitivity of the solution $\mathbf{x}$ to right hand side vector $\mathbf{b}$,
3. Sensitivity of $\mathbf{y} = A\mathbf{x}$ to the coefficient matrix A, and
4. Sensitivity of the solution $\mathbf{x}$ to the coefficient matrix A.

Problem 1

Sensitivity of $\mathbf{y}$ to a perturbation in $\mathbf{b}$.

In TB Lecture 12, the relative condition number is defined as

$$\kappa = \sup_{\delta x} \left(\frac{||\delta f||}{||f(x)||} \Big/ \frac{||\delta x||}{||x||} \right) \tag{2}$$

Problem 1(a)

Arguing directly from this definition, establish the condition number of $\mathbf{y}$ with respect to perburbations in $\mathbf{b}$ given by TB Lecture 18

$$\kappa = \frac{1}{\cos\theta} \tag{3}$$

Hint: The input "x" in this problem is $\mathbf{b}$ and the output (or model) "f" is $\mathbf{y}$. Show geometrically that the supremum is attained with $P\delta b = \delta b$.

Problem 1(b)

For $\theta = \pi/2$, the condition number is ∞. Illustrate what this means by considering the least squares problem

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix} [\, x \,] = \begin{bmatrix} -1 \\ 2 \end{bmatrix} \tag{4}$$

Use the results in TB 11.11 and 11.12 (page 82) to determine the projection operator P for this problem. Then compute $\mathbf{y} = P\mathbf{b}$ and show that $P\mathbf{b} = 0$. Find a perturbation $\delta\mathbf{b}$ so that $P\delta\mathbf{b} = \delta\mathbf{b} = \delta\mathbf{y} \neq 0$. Explain what a condition number $\kappa = \infty$ might mean here. Illustrate your argument graphically.

Problem 1(c)

Now consider the problem

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix} [\, x \,] = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \tag{5}$$

For this problem, show that $\kappa = 1$. What is qualitatively different about this problem than the problem in which $\kappa = \infty$?

Problem 2

Problem 18.1 in TB (page 136)

Problem 3

Show that if $(\lambda, \mathbf{v})$ is an eigenvalue/eigenvector pair for matrix A, then $((\lambda - \mu)^{-1}, \mathbf{v})$ is an eigenvalue/eigenvector pair for the matrix $(A - \mu I)^{-1}$.

Why is this observation useful when using the power iteration to find an eigenvalue close to μ?

Problem 4

Exercise 29.1 (Lecture 29, TB page 223). This is a five part problem that asks you to code an eigenvalue solver for a real, symmetric matrix using the shifted QR algorithm. Do your code in Python, using the Numpy `qr` algorithm where needed.

The basic steps are :

1. Reduce your matrix A to tridiagonal form. You may use the hessenberg code we wrote in class.

2. Implement the unshifted QR code (also done in class). Use the Numpy routine `qr`. Your iteration should stop when the off diagonal elements are smaller (in absolute value) than $\tau \approx 10^{-12}$.

3. Find all eigenvalues of a matrix A using the "deflation" idea described in Algorithm 28.2.

4. Introduce the Wilkinson shift, described in Lecture 29.

Notes

- Your code should work for a real, symmetrix matrix
- Your code does not have to be efficient in the sense of optimizing the cost of matrix/vector multiplies and so on.
- Apply your algorithm to the Hilbert matrix `scipy.linalg.hilbert`. The entries of the $m \times m$ Hilbert matrix are given by

$$H_{ij} = \frac{1}{i+j-1}, \qquad i,j = 1,2,\ldots m \tag{6}$$

In [11]:
```python
def display_mat(msg,A):
    print(msg)
    fstr = {'float' : "{:>10.6f}".format}
    with printoptions(formatter=fstr):
        display(A)
    print("")
```

In [12]:
```python
from scipy.linalg import hilbert

H = hilbert(5)
display_mat("Hilbert matrix : ", H)
```

```
Hilbert matrix :
array([[  1.000000,   0.500000,   0.333333,   0.250000,   0.200000],
       [  0.500000,   0.333333,   0.250000,   0.200000,   0.166667],
       [  0.333333,   0.250000,   0.200000,   0.166667,   0.142857],
       [  0.250000,   0.200000,   0.166667,   0.142857,   0.125000],
       [  0.200000,   0.166667,   0.142857,   0.125000,   0.111111]])
```

In [ ]: