# 08_eigenvalues

October 16, 2019

## 1 Eigenvalue and eigenvectors calculation

$$A\mathbf{x} = \lambda\mathbf{x}$$

#### 1.0.1 Power method (vector iteration)

- find the largest eigenvalue $\lambda_{max}$

$$\begin{align}
\mathbf{q}_k &= \frac{\mathbf{z}_{k-1}}{\|\mathbf{z}_{k-1}\|_2} \tag{1}\\
\mathbf{z}_k &= A\mathbf{q}_k \tag{2}\\
\lambda_{max}^k &= \mathbf{q}_k^T\mathbf{z}_k \tag{3}
\end{align}$$

[ ]:

```

```

[4]:

```
%matplotlib inline
from numpy import *
from matplotlib.pyplot import *
import numpy.linalg
import scipy.linalg

n = 9
h = 1./(n-1)

x=linspace(0,1,n)

a = -ones((n-1,))
b = 2*ones((n,))
A = (diag(a, -1) + diag(b, 0) + diag(a, +1))

A /= h**2

#print A
```

```
z0 = ones_like(x)


def PM(A,z0,tol=1e-5,nmax=500):
    q = z0/numpy.linalg.norm(z0,2)
    it = 0
    err = tol + 1.
    while it < nmax and err > tol:
        z = dot(A,q)
        l = dot(q.T,z)
        err = numpy.linalg.norm(z-l*q,2)
        q = z/numpy.linalg.norm(z,2)

        it += 1
    print("error =", err, "iterations =", it)
    print("lambda_max =", l)
    return l,q

l,x = PM(A,z0)

l_np, x_np = numpy.linalg.eig(A)

print("numpy")
print(l_np)
```

```
error = 8.45608648166e-06 iterations = 82
lambda_max = 249.735234086
numpy
[ 249.73523409  231.55417528  203.23651229  167.55417528  128.
    6.26476591   24.44582472   88.44582472   52.76348771]
```

#### 1.0.2 Inverse power method

- find the eigenvalue $\lambda$ **closest** to $\mu$

$$
\begin{aligned}
M &= A - \mu I && (4)\\
M &= LU && (5)\\
& && (6)\\
M\mathbf{x}_k &= \mathbf{q}_{k-1} && (7)\\
\mathbf{q}_k &= \frac{\mathbf{x}_k}{\|\mathbf{x}_k\|_2} && (8)\\
\mathbf{z}_k &= A\mathbf{q}_k && (9)\\
\lambda^k &= \mathbf{q}_k^T\mathbf{z}_k && (10)
\end{aligned}
$$

[9]:
```python
def IPM(A,x0,mu,tol=1e-5,nmax=500):
    pass # TODO

l,x = IPM(A,z0,6.)
```

```
error = 2.63101645873e-06 iterations = 3
lambda_max = 6.26476591422
```

[ ]:
```python

```