METHODS FOR COMPUTING EIGENVECTORS

1. Inverse Iteration Int. Atkinson 9.6 p.628 (Wielandt, 1944)

INVERSE ITERATION

Let \( A \) have eigenvalues \( \lambda_i \) and full set of \( n \)-vectors \( v_i \).
Let \( \lambda \) be a good approximation to \( \lambda_i \).

**PROCEDURE:**

\[
\begin{align*}
( A - \lambda I ) \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} &= \begin{bmatrix} x_0 \\ \vdots \\ x_0 \end{bmatrix}, \quad \text{arbitrary RHS}
\end{align*}
\]

We can write \( x_0 = \sum_{i=1}^n x_i v_i \).

The solution becomes

\[
\begin{align*}
x_i &= \sum_{i=1}^n \frac{x_i v_i}{\lambda_i - \lambda}.
\end{align*}
\]

Since \( \lambda \approx \lambda_i \), \( |x| \) is almost entirely a very large multiple of \( v_i \).

Normalize \( x_i \) and repeat with it as a new RHS, get \( x_2, \ldots \).

**TWO ISSUES:**

(i) Choice of \( x_0 \)

(ii) The linear system is very nearly singular \( \Rightarrow \) ill conditioned. Any danger?

(i). Almost any \( x_0 \) will do. Random vector works fine.

Wilkinson recommends \( ( A - \lambda I ) = PLU \); use \( \begin{bmatrix} u \mid x \end{bmatrix} = \begin{bmatrix} 1 \mid v \end{bmatrix} \).

(ii) Solution \( x_i \) is indeed badly determined (subject to small input noise), but only with regard to a scalar multiplier, which we do not care about anyway.

Heuristic argument why this is so:

\[
\begin{align*}
( A - \lambda I ) \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} &= \begin{bmatrix} x_0 \\ \vdots \\ x_0 \end{bmatrix} \\
\begin{bmatrix} A - \lambda I \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} &= \begin{bmatrix} x_0 \\ \vdots \\ x_0 \end{bmatrix}
\end{align*}
\]

Only the last element very small and lacking significant digits.
When doing back substitution, bottom element $y_j$ will be massively large - and uncertain. The rest of the back substitution will become essentially independent of the RHS. So $y_j$ will essentially be undetermined only with regard to the scalar multiplier introduced in its bottom entry.

**NEWTON’S METHOD**

Recall Newton for systems $f(x) = 0$:

(Atkinson, p. 108-111)

Iteration update $\Delta x_{i+1} = x_{i+1} - x_i$ obtained by solving linear system

$$\begin{bmatrix}
\frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \cdots & \frac{\partial f}{\partial x_n} \\
\frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \cdots & \frac{\partial f}{\partial x_n} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} & \cdots & \frac{\partial f}{\partial x_n}
\end{bmatrix}
\begin{bmatrix}
\Delta x_1 \\
\Delta x_2 \\
\vdots \\
\Delta x_n
\end{bmatrix}
= -
\begin{bmatrix}
f(x_1) \\
f(x_2) \\
\vdots \\
f(x_n)
\end{bmatrix}$$

Jacobian

Update

Residual

Nonlinear system to be solved:

$$Ax - \lambda x = 0$$

If we know $x$, want $\lambda$ : **linear, overdetermined**

If we know $\lambda$, want $x$ : **linear, singular**

Consider both $x$ and $\lambda$ as unknowns, then nonlinear $n$ equations, $n+1$ unknowns. Therefore add a normalization equation $x^* x = 1$.

The Newton system then becomes

$$\begin{bmatrix}
A - \lambda_i I & x_i \\
x_i & 0
\end{bmatrix}
\begin{bmatrix}
\Delta x_i \\
-\Delta \lambda_i
\end{bmatrix}
= -r_i$$

with

$$r_i = Ax_i - \lambda_i x_i$$

$$p_i = \frac{1}{2} (x_i^* x_i - 1)$$

Therefore iterate

$$\begin{cases}
x_{i+1} = x_i + \Delta x_i \\
\lambda_{i+1} = \lambda_i + \Delta \lambda_i
\end{cases} \quad i = 0, 1, 2, 3, \ldots$$
Some concluding observations on the Newton approach:

- Same matrix as in inverse iteration
- These col & row turn out to remove all ill-conditioning for system

+ The iterations not only give the e-vector, but also improve the e-value approximation

+ Quadratic convergence (rather than fast linear)

+ Excellent for following e-values/vectors when matrices change due to some parameter

+ Approach generalizes nicely from \((A-\lambda I)x = 0\) to \((A-xB)x = 0\) and \((A_1x + A_{r-1}x^{r-1} + \ldots + A_0x) = 0\).

- Slightly different coefficient matrix in each iteration - need to re-do LU-factorization
  (or use some quasi-Newton approach)