Computing Eigenvalues & Eigenvectors

Dominance and Real Eigenvalues

- It is often the case in applications that we do not require *all* eigenvalues for a given matrix.
- Sometimes we are only concerned with the *dominant eigenvalue* (if there is one) or simply a *largest eigenvalue*.
- We may also only be interested in *Real* eigenvalues.
- The *Perron-Frobenius Theorem* gives sufficient conditions on Real matrices that *guarantee* the existence of *positive Real largest* (but not necessarily *dominant*, ie *unique*) eigenvalues.
- These conditions also ensure existence of *positive Real eigenvectors* for the associated eigenvalue.

The Perron-Frobenius Conditions

- If A has only *non-negative Real* elements and is *irreducible*:

1. There is *unique positive Real* eigenvalue, λ_{pf}, associated with it (*unique* does not imply *dominant*).
2. This eigenvalue has an eigenvector, $\underline{x}_{pf}$, that contains only *positive Real components*.
3. Every other eigenvalue (*including Complex values*) is such that $|\lambda| \leq \lambda_{pf}$.

- *Irreducibility* is defined on p. 399 of course textbook.
- *Important in CS*: "the *adjacency matrix* of a *strongly-connected directed graph* is *irreducible*".
- A directed graph being *strongly-connected* if there is a *directed path* from *any node* to *any other node* in the graph.

The Power Method

- If A has a dominant eigenvalue the *Power Method* allows an eigenvector associated with it to be found.
- The method uses the fact that if $\underline{x}$ is an eigenvector of λ for A then *for any constant* $\alpha \in R$, $\alpha\underline{x}$ is *also* an eigenvector of λ.

1. Choose an initial "*guess*", eg $\underline{x}_0 = \langle 1,1,1,\ldots,1,1 \rangle$.
2. Set $i = 0$.
3. Compute $\underline{x}_{i+1} = A \cdot \underline{x}_i$
4. Set $i = i + 1$;
5. Continue from (3) **until** $i \geq MAX$. //* MAX is user-defined.

Problems with the Power Method

- Although the convergence to a dominant eigenvector is "*quick*" the *numbers* involved in computing successive $\underline{x}_i$ become "*large*" so *increasing the computation time* needed.
- This problem is easily overcome: use "*scaling*".
- *The Power Method with Scaling* adds the step:

$$\underline{x}_i := \frac{\underline{x}_i}{\max\{|x_k| : x_k \in \underline{x}_i\}}$$

after Step 4.

- This *does not affect the result* (the vector is replaced with a *scalar multiple* of itself) and ensures the numbers are "*small*".

Eigenvalues – Rayleigh Quotient

- We can find (approximations to) dominant eigenvectors.
- What about their associated *eigenvalue*?
- The concept of *Rayleigh Quotient* allows us to find this.
- Suppose we have found (eg by the Power Method) the approximating eigenvector $\underline{x}$.
- The *Rayleigh Quotient* is:

$$\frac{(A \cdot \underline{x}) \cdot \underline{x}^{\mathrm{T}}}{\underline{x} \cdot \underline{x}}$$

- This uses the *dot product* and reports the *eigenvalue* for $\underline{x}$.

Other Cases

- We can find the *smallest eigenvalue* of A if it is *non-singular*.
- The *smallest eigenvalue* of A is $\frac{1}{\lambda}$ where λ is the *largest eigenvalue* of A^{-1}.
- Techniques such as *The Inverse Power Method* (page 409) and *Deflation* (pp. 409–413) allow other eigenvalues to be found.
- *Jacobi's Algorithm* is a sophisticated technique for extracting representative information for *symmetric matrices*.
- In practice many languages have support software packages that can be applied: numpy in Python; JAMA in Java.

Where other eigenvalues arise

- The two computational applications we will look at use, in the first case, dominant eigenvalue and an associated eigenvector.
- In the second case a way of describing any matrix in terms of a product of three matrices with only Real eigenvalues.
- The first is *Google's Page Rank Algorithm*.
- The second a technique called *Singular Value Decomposition*.
- This arises in extracting important information from *large data pools* (in *Data Science*) and refining *Machine Learning*.
- The use we look at is in *Image Compression*.

Spectral Graph Theory

- The relationship between the *spectra* of (adjacency matrices) of graphs and *properties of the graphs* themselves has been a long established study in *Algorithmics* and *Computer Science*.
- The *number of colours* needed properly to colour the nodes of a graph so that no two linked nodes have the same colour is *at most* $\lambda_1(G) + 1$ and *at least* $1 + \frac{\lambda_1(G)}{|\lambda_n(G)|}$.
- The *number of links* is *exactly*

$$\frac{1}{2}\sum_{k=1}^{n}(\lambda_k(G))^2$$

Summary

- Spectral analysis of graphs, networks and matrix (table) structures is an area of *increasing importance in modern CS*.
- It has been widely-used in approaches to "*ranking problems*" where the aim is to find the "*best ordering*" of some collection of items, eg *web pages*, *predictive ordering of sports leagues*, "*strength of argument*".
- It is applied in *Machine Learning* and *Data Science* studies.
- It has led to deep results in *Graph Algorithmics* and the field of *Computational Game Theory*.
- We turn to one of the *most significant successes* in the next lecture: *Google's Page Rank algorithm*.