On Generalized Arnoldi Methods for the Computation of PageRank

JunFeng Yin

Tongji University
Shanghai, P.R. China

Joint work with GuoJian Yin.

JSIAM Spring meeting, Tsucuba University, Mar. 8-9, 2010
Outline

1. Web Search and PageRank
   - The Google matrix

2. Power method and Acceleration Techniques
   - Power method
   - Acceleration Techniques

3. Arnoldi Based Methods
   - A refined Arnoldi method
   - Generalized Arnoldi method
   - Numerical experiments
Web Search and Pagerank

- An important problem in web search is to classify the pages in order according to their importance.
- How to define the "importance"?

**Definition**

A page is important if other important pages point to it.

- Based on this idea, S. Brin & L. Page (1998) proposed the PageRank algorithm which is the beginning of Google.

**Advantage**

Importance of a webpage is determined not by its contents but rather by which pages link to it.
The PageRank model:

**Definition**

Given a Webpage database, the PageRank of the $i$th Webpage is the $i$th element $\pi_i$ of $\pi$, which satisfies

$$\pi^T P = \pi^T$$

where $P$ is a matrix of weights of webpages that indicate their importance.

- **The Google matrix:** $P$, is nonnegative, defined by the hyperlink structure.
- **The Pagerank vector:** $\pi$, is probability vector satisfied $\pi^T \mathbf{e} = 1$, where $\mathbf{e}$ is a vector of all-ones.
The Google Matrices

Definition
Let \( d_i \geq 1 \) be the outdegree of the page \( i \), that is the number of pages pointing to it. The Google matrix \( P = \{p_{ij}\} \) is defined by

\[
p_{ij} = \begin{cases} 
0, & \text{If there is no outlink from page } i \text{ to page } j \\
0, & \text{If } i = j; \\
1/d_i, & \text{Otherwise.}
\end{cases}
\]

Difficulties

- \( P \) is too large (size possibly in the billions) for forming any of our favorite decompositions.
- \( P \) could be reducible, contain zero rows, and other difficulties of this sort.

How do we modify \( P \) so that there is a unique solution?
The existence of **dangling nodes** (correspond to an all-zero row in the matrix), which could have very important pages but without any outlinks.

For examples,

\[
P = \begin{pmatrix}
0 & .5 & .5 \\
.5 & 0 & .5 \\
.5 & 0 & 0 \\
0 & 0 & 0
\end{pmatrix}
\]

★ Here, \( P \) is **not** row-stochastic, since row 3 contain all zeros.
★ The remedy is to replace all zero rows with $\frac{1}{n}e^T$, where $e^T$ is the row vector of all ones and $n$ is the order of $P$.

- For instance,

\[
\tilde{P} = \begin{pmatrix}
0 & .5 & .5 \\
.5 & 0 & .5 \\
.3 & .3 & .3
\end{pmatrix}
\]

$\tilde{P}$ is stochastic with dominant eigenvalue 1 and $\tilde{P}e = e$. 
An important Theorem

We are seeking a row vector $\pi^T$ where $P$ is a square stochastic matrix, with nonnegative entries between 0 and 1, and $Pe = e$.

**Theorem**

*Perron(1907)-Frobenius(1912)*: A nonnegative irreducible matrix has a simple real eigenvalue equal to its spectral radius, whose associated eigenvector is a vector all of whose entries are nonnegative.

What happens when $P$ is stochastic and possible reducible?
Second Difficulties

Difficulties

**Periodicity**: a cyclic path in the Webgraph. (e.g. one points only to his wife’s webpage and his wife points only to his.)

Simple examples,

\[ \tilde{P} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \]

★ Here, \( \tilde{P} \) can have several eigenvalues on the unite circle, thus causing convergence problems to power method.
Solution

\( \tilde{P} \) is replaced by

\[
P_\alpha = \alpha \tilde{P} + (1 - \alpha) \nu v^T
\]

with \( \alpha \in [0, 1) \), and \( \nu \) a probability vector satisfied \( \nu \geq 0 \) and \( \nu^T e = 1 \).

- \( P_\alpha \) is stochastic and irreducible.
- It has 1 as its dominant eigenvalue with \( e \) as its corresponding right eigenvector.
We have $P_\alpha > 0$ which yields a unique solution. But what is the significance of the stationary probability vector?

$P_\alpha$ is a Markov chain with positive entries, and

$$P_\alpha z = z$$

Therefore for $\alpha < 1$, $z$ is unique (under proper scaling).
For the identity matrix, $P = I$, no unique stationary probability distribution.

But for $P_\alpha = \alpha I + (1 - \alpha) ee^T / n$, we have

$$z = \frac{1}{n} e$$

and

$$\pi = e$$
The Eigenvalues of the PageRank Matrix

[ Elden, 2003; Langville & Meyer 2003; Serra-Capizzano, 2005 ]

**Theorem**

Let \( P \) be a row-stochastic matrix with eigenvalues \( \{1, \lambda_2, \lambda_3, \ldots, \lambda_n\} \).

Then the eigenvalues of \( P_\alpha \), where \( 0 < \alpha < 1 \) and \( \nu \) is a probability vector, are

\[
\{1, \alpha \lambda_2, \alpha \lambda_3, \ldots, \alpha \lambda_n\}
\]

This implies

\[
\frac{|\lambda_i|}{|\lambda_q|} \leq \alpha, \quad i = 2, 3, \ldots, n
\]
In the following, denote $A = P^T_\alpha$ column-stochastic, and $x = \pi$ be the PageRank vector we wanted.

In Brin & Page’s model, the eigenvalue problem is

$$x = Ax$$

and they use the power method

$$x^{(k+1)} = Ax^{(k)}$$
Assume that $Au_i = \lambda_i u_i$. For $|\lambda_i| \neq |\lambda_j|$, we have

$$x^{(0)} = \sum \alpha_i u_i$$

and

$$x^{(k)} = \sum \alpha_i \lambda_i^k u_i$$

with $\|x^{(k)}\|_1 = 1$ and $x \geq 0$.

After normalization, for $\lambda_1 = 1$, we have

$$x^{(k)} = u_1 + \sum_{i=2} \beta_i \lambda_i^k u_i$$

Note that $|\lambda_2| \leq \alpha$, which influences the convergence.
Extrapolation

Idea

Slowly convergent series can be replaced by series that converge to the same limit at a much faster rate.
Extrapolation

- Aitken Extrapolation & Quadratic Extrapolation.  
  (Kamvar, Haveliwala, Manning & Golub, 2003)
- $A^d$ Extrapolation.  
  (Haveliwala, Kamavar, Klein, Manning & Golub,)
- Minimal polynomial Extrapolation & reduced rank extrapolation.  
  (Sidi, 2008)
- Polynomial Extrapolation & Least square Extrapolation.  
  (Brezinski & Redivo-Zaglia, 2006)
- The performance of extrapolation is not good if it is applied in every step. Usually, extrapolation is taken every 10 iterations.
Other Effective Approaches

- Aggregation/Disaggregation.
  (Stewart, Langville & Meyer, ...)

- Approaches related to permutations of the Google matrix.
  (Del Corso et. al., Kamvar et. al.)

- Linear system formulation.
  (Arasu et. al.)

- Arnoldi based approach.
  (Golub & Greif, Wu & Wei.)
Arnoldi Process

- Given \( A \in \mathbb{R}^{n \times n} \) and \( q_0 \in \mathbb{R}^n \), compute a sequence of orthonormal basis

\[
Q_k = [q_0, q_1, \ldots, q_{k-1}]
\]

of Krylov subspace

\[
\mathcal{K}_k(A, v_0) = \text{span}\{q_0, Aq_0, A^2q_0, \ldots, A^{k-1}q_0\}
\]

so that

\[
AQ_k = Q_{k+1}H_{k+1,k}, \quad Q_k^T AQ_k = H_{k,k}
\]

where \( H_{k+1,k} \in \mathbb{R}^{k+1 \times k} \) is upper Hessenberg matrix.
Arnoldi method for eigenvalue problem

- Let $\lambda$ and $x$ be the eigenvalue and eigenvector of $A$,

$$Q_n^T A Q_n (Q_n^T x) = \lambda (Q_n^T x)$$

- Let $H_{k,k} = Q_k^T A Q_k$, also upper Hessenberg. The eigenvalue of $H_{k,k}$ denoted by $\{\theta_i\}_{i=1}^{k}$, called Ritz values of $A$.

Restarted Arnoldi method for eigenvalue problem

For $i = 1, 2, \ldots$ until convergence

Compute $[Q_k, H_{k,k}] = \text{Arnoldi}(A, q, k)$,

Compute $H_{k,k} v_M = \theta_{\text{max}} v_M$,

Set $q = Q_k v_M$,

endFor
Arnoldi method for eigenvalue problem

Main Idea
- The Arnoldi method is generally used for generating a small upper Hessenberg that approximates some of the eigenvalues of the original matrix.
- Main cost: One matrix-vector product (with original large matrix) per iteration, Inner products and norm computations.

Advantage
\[ A \in \mathbb{R}^{n \times n} \text{ is very Huge, but } H \text{ is } k \times k \text{ with } k \ll n. \]

Disadvantage
The cost of inner products and norm computations is expensive as \( k \) increases.
Arnoldi based method for PageRank

   Golub & Greif.

   Gang Wu & Yimin Wei.

3. Arnoldi-extrapolation algorithm, 2008  
   Gang Wu & Yimin Wei.

4. Generalized Arnoldi algorithm  
   JunFeng Yin & GuoJian Yin.
In order to improve the accuracy of the eigenvector $v_M$

$$H_{k,k}v_M = \theta_{\text{max}}v_M,$$

a refined Arnoldi method was considered by

$$\min \| (H_{k+1,k} - \theta_{\text{max}} \begin{pmatrix} I_k \\ 0 \end{pmatrix})v_M \|$$

Here, $v_M$ is called refined Ritz vector.

The idea and convergence analysis from [Jia, 1997; Jia & Stewart, 2001].
Algorithm

1. For $i = 1, 2, \ldots$ until convergence
2. Compute $[Q_k, H_{k,k}] = \text{Arnoldi}(A, q, k)$,
3. Compute $H_{k,k}v_M = \theta_{\text{max}}v_M$,
4. Compute $H_{k+1,k} - \theta_{\text{max}}[I \ 0]^T = U\Sigma V^T$,
5. Choose refined Ritz vector $v_k$ corresponding to $\sigma_{\text{min}}$,
6. Set $q = Q_k v_k$,
7. endFor
Refined Arnoldi method for PageRank problem

**Advantage**

For PageRank problem, take the largest Ritz value is known to be 1, the cost of the algorithm can be greatly saved.

**Algorithm**

For $i = 1, 2, \ldots$ until convergence

- Compute $[Q_k, H_{k,k}] = \text{Arnoldi}(A, q, k)$,
- Compute $H_{k+1,k} - [I \ 0]^T = U \Sigma V^T$,
- Choose refined Ritz vector $v_k$ corresponding to $\sigma_{\text{min}}$,
- Set $q = Q_k v_k$,

endFor
Stopping criterion

Since

\[ Aq - q = AQ_k v_k - Q_k v_k \]
\[ = Q_{k+1} H_{k+1,k} v_k - Q_k v_k \]
\[ = Q_{k+1} (H_{k+1,k} - \begin{pmatrix} I_k & 0 \end{pmatrix}) v_k \]
\[ = Q_{k+1} U \Sigma V^T v_k = \sigma_{\min} Q_{k+1} u_k \]

we have

\[ \| Aq - q \|_2 = \sigma_{\min} \]

Thus, \( \sigma_{\min} \) can be used in stopping criterion without additional computation for the residual.
Advantage

1. Orthogonalization achieves effective separation of eigenvectors.
2. Take advantage of knowing the largest eigenvalue.
3. Largest Ritz value could be complex, but if we set the shift to 1 then no risk of complex arithmetic.
4. Smallest singular value converges smoothly to zero (more smoothly than largest Ritz value converges to 1).
5. Stopping criterion with no computational overhead:

\[ \| Aq - q \|_2 = \sigma_{\text{min}}(H - [I; 0]). \]
Disadvantages

1. More complicated to implement.

2. A single iteration is more expensive than a power iteration; must converge within fewer iterations.

Thus, how to accelerate this Arnoldi-type method?
Let $G$ be SPD, define an inner product and a norm as

$$<u, v>_G = u^T G v = \sum_{i=1}^{n} \sum_{j=1}^{n} g_{ij} u_i v_j$$

$$\|u\|_G = \sqrt{<u, u>_G} = \sqrt{u^T G u}$$

General Arnoldi method leads to

$$\tilde{Q}_k^T G \tilde{Q}_k = I, \quad A\tilde{Q}_k = \tilde{Q}_{k+1} \tilde{H}_{k+1,k}$$

where the columns of $\tilde{Q}_k$ are $G$ orthogonal.

When $G = I$, it leads to standard Arnoldi method.
Difference between two Arnoldi processes

\[ \tilde{Q}_k G A \tilde{Q}_k = \tilde{H}_{k,k} \]

- **General Arnoldi method:**

  \[
  \min_{q \in \mathbb{R}^n} \| Aq - q \|_G = \min_{q \in \mathbb{R}^n} (Aq - q)^T G (Aq - q)
  \]

- **Left preconditioned Arnoldi method:**

  \[
  \min_{q \in \mathbb{R}^n} \| G Aq - q \|_2 = \min_{q \in \mathbb{R}^n} (G Aq - q)^T (G Aq - q)
  \]
Residual vector for Pagerank

Since

$$\begin{align*}
Aq - q &= A\tilde{Q}_k v_k - \tilde{Q}_k v_k \\
     &= \tilde{Q}_{k+1} \tilde{H}_{k+1,k} v_k - \tilde{Q}_k v_k \\
     &= \tilde{Q}_{k+1}(\tilde{H}_{k+1,k} - \begin{pmatrix} I_k \\ 0 \end{pmatrix}) v_k \\
     &= \tilde{Q}_{k+1} U \Sigma V^T v_k = \sigma_{\min} \tilde{Q}_{k+1} u_k
\end{align*}$$

we have

$$\|Aq - q\|_G = \sigma_{\min}$$
Question
How to accelerate the convergence the Arnoldi-type method for Pagerank computation?

- One approach to enlarge the weight of the residual components who converge slowly.
- Another approach to apply Arnoldi method adaptively to the pages converged slowly.
Observation of residual

- Denote

\[ r = Aq - q = \sigma_{\min} \tilde{Q}_{k+1} u_k \]

It include the convergence information of the components of the residual.

- Moreover, it tell us the convergence velocity of every component of the residual.
Choice of $G$

We strengthen the weight of the residual components who converge slowly by the information from the residual.

For instance,

$$G = \text{diag}\left\{ \frac{|r|}{\|r\|_2} \right\}$$

Moreover, matrix $G$ can be defined adaptively by the residual in every loop.
Computational environment

- Initial vector $q_0 = e$.
- $k = 5$ in Arnoldi process.
- Stopping rule is
  \[ \|Aq - q\|_1 \leq 1.0 \times 10^{-8} \]
- Dell machine with 3.0 GHz CPU and 2G memory.
First example

Example

* Hollins.dat

★ generated from a web crawl of www.hollins.edu,
★ contains 6,012 nodes and 23,875 links.
The number of iteration steps and CPU time

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>Power</th>
<th>QE-Power</th>
<th>Arnoldi</th>
<th>G-Arnoldi</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.85</td>
<td>92</td>
<td>63</td>
<td>65</td>
<td>52</td>
</tr>
<tr>
<td>0.90</td>
<td>140</td>
<td>88</td>
<td>90</td>
<td>69</td>
</tr>
<tr>
<td>0.95</td>
<td>285</td>
<td>147</td>
<td>140</td>
<td>98</td>
</tr>
<tr>
<td>0.99</td>
<td>1438</td>
<td>581</td>
<td>523</td>
<td>220</td>
</tr>
<tr>
<td>0.995</td>
<td>2870</td>
<td>1419</td>
<td>1022</td>
<td>313</td>
</tr>
<tr>
<td>0.998</td>
<td>7148</td>
<td>2322</td>
<td>1235</td>
<td>491</td>
</tr>
</tbody>
</table>
The norm of residual versus iteration number

\[ \text{Iterations} \]

\[ \text{Residual norms} \]

\[ \alpha = 0.85 \]

\[ \text{Power}, \text{QE-Power}, \text{Arnoldi}, \text{G-Arnoldi} \]

\[ \alpha = 0.95 \]

\[ \text{Iterations} \]

\[ \text{Residual norms} \]

\[ \alpha = 0.99 \]

\[ \text{Iterations} \]

\[ \text{Residual norms} \]

\[ \alpha = 0.995 \]

\[ \text{Iterations} \]

\[ \text{Residual norms} \]
Example

*Stanford.dat*

★ generated from a crawl of the Stanford.edu domain by the Stanford Web-Base project.

★ contains roughly 281,903 nodes with 2,312,497 links.
The number of iteration steps and CPU time

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>Power</th>
<th>QE-Power</th>
<th>Arnoldi</th>
<th>G-Arnoldi</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.85</td>
<td>95</td>
<td>81</td>
<td>90</td>
<td>78</td>
</tr>
<tr>
<td>0.90</td>
<td>146</td>
<td>121</td>
<td>125</td>
<td>109</td>
</tr>
<tr>
<td>0.95</td>
<td>299</td>
<td>231</td>
<td>233</td>
<td>168</td>
</tr>
<tr>
<td>0.99</td>
<td>1512</td>
<td>1019</td>
<td>864</td>
<td>546</td>
</tr>
<tr>
<td>0.995</td>
<td>3001</td>
<td>1901</td>
<td>1552</td>
<td>793</td>
</tr>
<tr>
<td>0.998</td>
<td>7401</td>
<td>3906</td>
<td>3310</td>
<td>1713</td>
</tr>
</tbody>
</table>
The norm of residual versus iteration number

\begin{itemize}
    \item \textbf{alpha=0.85}
    \item \textbf{alpha=0.95}
    \item \textbf{alpha=0.99}
    \item \textbf{alpha=0.995}
\end{itemize}
Discussion

- G-Arnoldi method can greatly reduce the number of iteration.

- For $\alpha$ is relatively small, $\alpha = 0.85$, QE-Power is faster; while for $\alpha$ is large, e.g., $\alpha = 0.99$, G-Arnoldi is faster.

- As the restart $k$ increasing, the number of matrix-vector multiplication is reduced while the computation for inner product and orthogonalization is increasing.

- Practical experiments show that $k = 5$ is best in terms of total CPU time.
The norm of residual versus iteration number

Hollins.dat

![Graphs showing residual norms for different values of alpha and iteration numbers.](image)
Short name

- **P**: Power method.
- **A**: Arnoldi-type method.
- **GA**: Generalized Arnoldi method.
- **PGA**: Power combined Generalized Arnoldi method.
The number of iteration steps and CPU time

**Hollins.dat**

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>IT</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>A</td>
</tr>
<tr>
<td>0.85</td>
<td>92</td>
<td>45</td>
</tr>
<tr>
<td>0.90</td>
<td>140</td>
<td>57</td>
</tr>
<tr>
<td>0.95</td>
<td>285</td>
<td>89</td>
</tr>
<tr>
<td>0.99</td>
<td>1438</td>
<td>203</td>
</tr>
<tr>
<td>0.995</td>
<td>2870</td>
<td>338</td>
</tr>
<tr>
<td>0.998</td>
<td>7148</td>
<td>473</td>
</tr>
</tbody>
</table>
### Stanford.dat

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>IT</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>A</td>
</tr>
<tr>
<td>0.85</td>
<td>95</td>
<td>75</td>
</tr>
<tr>
<td>0.90</td>
<td>146</td>
<td>102</td>
</tr>
<tr>
<td>0.95</td>
<td>299</td>
<td>166</td>
</tr>
<tr>
<td>0.99</td>
<td>1512</td>
<td>577</td>
</tr>
<tr>
<td>0.995</td>
<td>3001</td>
<td>718</td>
</tr>
<tr>
<td>0.998</td>
<td>7401</td>
<td>1855</td>
</tr>
</tbody>
</table>
Discussion

- PGA method can take the advantage of both two methods.
- The converge speed of PGA method is similar to GA method, but the CPU time is obviously smaller than GA method.
- Also, the converge speed of PGA method is obviously faster than Power method, then the CPU time the former is obviously small.
Summary

- Power method and Arnoldi approach are effective.
- Our techniques can accelerate the convergence evidently, especially for large $\alpha$.

Challenges

- Theoretical analysis for the convergence of G-Arnoldi and the optimal choice of the matrix $G$.
- Efficient methods for a large value of $\alpha$.​
A survey of eigenvector methods of Web information retrieval.
(Lagville & Meyer, 2005)

The PageRank vector: properties, computation, approximation and acceleration.
(Brezinski & Redivo-zaglia, 2006)

(Pavel Berkhin, 2005)
Thank you!