Model Reduction for Edge-Weighted Personalized PageRank

D. Bindel

2 Dec 2016
The Computational Science & Engineering Picture

- MEMS
- Smart grids
- **Networks**
- Systems

- **Linear algebra**
- Approximation theory
- Symmetry + structure
- Optimization

- HPC / cloud
- Simulators
- **Solvers**
- Frameworks
Collaborators

Wenlei Xie
David Bindel
Johannes Gehrke
Al Demers

LinkedIn
Cornell
Microsoft
Cornell
PageRank Problem

Goal: Find “important” vertices in a network

- Basic approach uses only topology
- Weights incorporate prior info about important nodes/edges
PageRank Model

- Random surfer model: \( x^{(t+1)} = \alpha P x^{(t)} + (1 - \alpha) v \) where \( P = AD^{-1} \)
- Stationary distribution: \( M x = b \) where \( M = (I - \alpha P) \), \( b = (1 - \alpha) v \)
Introduce *personalization parameters* $w \in \mathbb{R}^d$ in two ways:

- **Node weights:** $M x(w) = b(w)$
- **Edge weights:** $M(w) x(w) = b$
Node weight personalization is well-studied

- Topic-sensitive PageRank: fast methods based on linearity
- Localized PageRank: fast methods based on sparsity

Some work on edge weight personalization

- ObjectRank/ScaleRank: personalize weights for different edge types
- But lots of work incorporates edge weights *without* personalization

**Our goal**: General, fast methods for edge weight personalization
Edge Weight Parameterizations

Different ways to personalize ⟷ different algorithm options

1. **Linear**: Take an edge of type $i$ with probability $\alpha w_i$

   \[ P(w) = \sum_{i=1}^{d} w_i P(i) \]

2. **Scaled linear**: Take an edge with probability $\propto (\text{linear})$ edge weight

   \[ P(w) = A(w)D(w)^{-1}, \quad A(w) = \sum_{i=1}^{d} w_i A(i), \quad D(w) = \sum_{i=1}^{d} w_i D(i), \]

3. **Fully nonlinear**: Both $A$ and $P$ depend nonlinearly on $w$
Model Reduction

Expensive full model \((Mx = b)\)

\[ \approx \]

Reduced basis \(U\)

Reduced model \((\tilde{M}y = \tilde{b})\)

\(\text{Approximation ansatz}\)

Model reduction procedure from physical simulation world:

- **Offline**: Construct reduced basis \(U \in \mathbb{R}^{n \times k}\)
- **Offline**: Choose \(\geq k\) equations to pick approximation \(\hat{x} = Uy\)
- **Online**: Solve for \(y(w)\) given \(w\) and reconstruct \(\hat{x}\)
Reduced Basis Construction: SVD (aka POD/PCA/KL)

Snapshot matrix

\[ \begin{pmatrix} x_1 & x_2 & \cdots & x_r \end{pmatrix} \approx U \Sigma V^T \]

Sample points
Choosing Good Spaces

What is the best possible approximation $\hat{x} = Uy$?

$$\min_y \| Uy - x(w) \|_2 \leq \sigma_{k+1} \| x \|_2 + e_{\text{interp}}(w)$$

where

$$e_{\text{interp}}(w) = \left\| x(w) - \sum_{j=1}^{r} x(w_j) c_j(w) \right\|_2$$

is error in an interpolant.

- Pay attention where $x$ has large derivatives!
- Also suggests sampling strategies (sparse grids, adaptive methods)
Approximation Ansatz

Want \( r = MUy - b \approx 0 \). Consider two approximation conditions:

<table>
<thead>
<tr>
<th>Method</th>
<th>Ansatz</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bubnov-Galerkin</td>
<td>( U^T r = 0 )</td>
<td>Good accuracy empirically&lt;br&gt;Fast for ( P(w) ) linear</td>
</tr>
<tr>
<td>DEIM (collocation)</td>
<td>( \min | r_I | )</td>
<td>Fast even for nonlinear ( P(w) )&lt;br&gt;Complex cost/accuracy tradeoff</td>
</tr>
</tbody>
</table>

Petrov-Galerkin a bit more accurate than Bubnov-Galerkin – future work.
Bubnov-Galerkin Method

Linear case: $w_i = \text{probability of transition with edge type } i$

$$M(w) = I - \alpha \left( \sum_i w_i P^{(i)} \right) , \quad \tilde{M}(w) = I - \alpha \left( \sum_i w_i \tilde{P}^{(i)} \right)$$

where we can precompute $\tilde{P}^{(i)} = U^T P^{(i)} U$

Nonlinear: Cost to form $\tilde{M}(w)$ comparable to cost of PageRank!
Discrete Empirical Interpolation Method (DEIM)

\[ y - M U - b = 0. \]

Equations in \( \mathcal{I} \)

- Ansatz: Minimize \( \| r_\mathcal{I} \| \) for chosen indices \( \mathcal{I} \)
- Only need a few rows of \( M \) (and associated rows of \( U \))
  - If given \( A(w) \), also need column sums for normalization.
- Difference from physics applications: high-degree nodes!
Similar error analysis framework for both Galerkin and DEIM

\[ \text{Consistency} + \text{Stability} = \text{Accuracy} \]

- **Consistency**: Does the subspace contain good approximants?
- **Stability**: Is the approximation subproblem far from singular?

Characterize stability by a quasi-optimality condition

\[ \|x - Uy\| \leq \min_z C \|x - Uz\| \]
Standard Quasi-Optimality Approach

- Define a solution projector:

  \[ \Pi x = \text{approximate solution when true solution is } x \]

  Note that \( \Pi U = U \).

- The error projector \( I - \Pi \) maps a true solution to error

  \[ e = x - \Pi x = (I - \Pi)x \]

  Note that \( (I - \Pi)U = 0 \).

- If \( e_{\text{min}} = x - Uz \) is the smallest norm error in the space, then

  \[ e = (I - \Pi)x - (I - \Pi)Uz = (I - \Pi)e_{\text{min}} \]

  Therefore, a bound on \( \| I - \Pi \| \leq 1 + \| \Pi \| \) establishes quasi-optimality.
Quasi-Optimality: Galerkin and DEIM

\[
\begin{align*}
\text{Galerkin:} & \quad \Pi = U \tilde{M}^{-1} W^T M \\
\text{DEIM:} & \quad \Pi = U \tilde{M}^\dagger M_{\mathcal{I},:} \\
\tilde{M} & \equiv W^T M U
\end{align*}
\]

- Key to stability: \( \tilde{M} \) far from singular
- Suggests pivoting schemes for “good” \( \mathcal{I} \) in DEIM
  - Also helps to explicitly enforce \( \sum_i \hat{x}_i = 1 \)
- Can bound \( \|\Pi\| \) offline for Galerkin + linear parameterization.
Interpolation Costs

Consider subgraph relevant to one interpolation equation:

- Really care about weights of edges incident on $\mathcal{I}$
- Need more edges to normalize (unless $A(w)$ linear)
- Cost to include $i \in \mathcal{I}$: $|\{j, k : a_{ij} \neq 0 \text{ and } a_{kj} \neq 0\}|$
- High in/out degree are expensive but informative
Key question: how to choose $\mathcal{I}$ to balance cost vs accuracy?

Want to pick $\mathcal{I}$ once, so look at rows of

$$Z = [M(w_1)U \ M(w_2)U \ldots]$$

for sample parameters $w^{(i)}$.

Pivoted QR-like greedy row selection with proxy measures for

- **Cost:** Nonzeros in row (+ assoc columns if normalization required)
- **Accuracy:** Residual when projecting row onto those previously selected

Several heuristics for cost/accuracy tradeoff (see paper)
If $\ell = \#$ PR components needed, online costs are:

- Form $\tilde{M}$ \(O(dk^2)\) for B-G
- More complex for DEIM
- Factor $\tilde{M}$ \(O(k^3)\)
- Solve for $y$ \(O(k^2)\)
- Form $Uy$ \(O(k\ell)\)

Online costs do not depend on graph size! (unless you want the whole PR vector)
Example Networks

DBLP (citation network)
- 3.5M nodes / 18.5M edges
- Seven edge types $\Rightarrow$ seven parameters
- $P(w)$ linear
- Competition: ScaleRank

Weibo (micro-blogging)
- 1.9M nodes / 50.7M edges
- Weight edges by topical similarity of posts
- Number of parameters = number of topics (5, 10, 20)

(Studied global and local PageRank – see paper for latter.)
Singular Value Decay

\[ r = 1000 \text{ samples, } k = 100 \]
DBLP Accuracy

- Kendall@100
- Normalized L1

- Galerkin
- DEIM-100
- DEIM-120
- DEIM-200
- ScaleRank

D. Bindel
NYU
2 Dec 2016
DBLP Running Times (All Nodes)

- **Coefficients**
- **Construction**

Running time (s):
- Galerkin
- DEIM-100
- DEIM-120
- DEIM-200
- ScaleRank

<table>
<thead>
<tr>
<th>Method</th>
<th>Running Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Galerkin</td>
<td>0.4</td>
</tr>
<tr>
<td>DEIM-100</td>
<td>0.4</td>
</tr>
<tr>
<td>DEIM-120</td>
<td>0.4</td>
</tr>
<tr>
<td>DEIM-200</td>
<td>0.4</td>
</tr>
<tr>
<td>ScaleRank</td>
<td>0.7</td>
</tr>
</tbody>
</table>
Weibo Accuracy

![Bar Chart]

- **Kendall@100**
- **Normalized L1**

### Parameters
- 5
- 10
- 20

D. Bindel  
NYU  
2 Dec 2016  
26 / 40
Application: Learning to Rank

Goal: Given $T = \{(i_q, j_q)\}_{q=1}^T$, find $w$ that mostly ranks $i_q$ over $j_1$.
(c.f. Backstrom and Leskovec, WSDM 2011)

Standard idea: Gradient descent

$$\frac{\partial x}{\partial w_j} = M(w)^{-1} \left[ \alpha \frac{\partial P(w)}{\partial w_j} x(w) \right]$$

Dominant cost: $d + 1$ solves with the PageRank system $M(w)$

- One PageRank solve to evaluate a loss function
- One PageRank solve per parameter to evaluate gradients
Goal: Given $T = \{(i_q, j_q)\}_{q=1}^{\frac{|T|}{q}}$, find $w$ that mostly ranks $i_q$ over $j_1$. (c.f. Backstrom and Leskovec, WSDM 2011)

- **Standard**: Gradient descent on full problem
  - One PR computation for objective
  - One PR computation for each gradient component
  - Costs $d + 1$ PR computations per step

- **With model reduction**
  - Rephrase objective in reduced coordinate space
  - Use factorization to solve PR for objective
  - Re-use same factorization for gradient
DBLP Learning Task

![Graph showing the comparison of Standard, Galerkin, and DEIM-200 methods in terms of objective function value over iterations. The graph shows a descending trend with markers for each iteration.](image)

(8 papers for training + 7 params)
The Punchline

Test case: DBLP, 3.5M nodes, 18.5M edges, 7 params

Cost per Iteration:

<table>
<thead>
<tr>
<th>Method</th>
<th>Standard</th>
<th>Bubnov-Galerkin</th>
<th>DEIM-200</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time (sec)</td>
<td>159.3</td>
<td>0.002</td>
<td>0.033</td>
</tr>
</tbody>
</table>
In the paper (but not the talk)

- Selecting interpolation equations for DEIM
- Localized PageRank experiments (Weibo and DBLP)
- Comparison to BCA for localized PageRank

**Room for future work!** Analysis, applications, systems, ...
Questions?

Edge-Weighted Personalized PageRank: Breaking a Decade-Old Performance Barrier

Wenlei Xie, David Bindel, Johannes Gehrke, and Al Demers

KDD 2015, paper 117

Sponsors:

- NSF (IIS-0911036 and IIS-1012593)
- iAd Project from the National Research Council of Norway
Trailers!
Spectral Topic Modeling

\[ C \approx B \times A \times B^T \]
Music of the Microspheres
Fast Fingerprints for Power Systems
Response Surfaces for Global Optimization

Six-hump camel function

[Image of a 2D contour plot representing a six-hump camel function with various contour lines and data points.]
Graph Densities of States
http://www.cs.cornell.edu/~bindel