Rational extrapolation for the PageRank vector

Michela REDIVO ZAGLIA
University of Padova - Italy

joint work with

Claude BREZINSKI
University of Lille - France

- **The Google matrices**
- **Extrapolation**
- **Numerical experiments**

THE GOOGLE MATRICES

Let $\mathbf{deg(i)} > \mathbf{0}$ be the **outdegree** of the page $\mathbf{i}$, that is the number of pages it points to.

The **Google matrix** $\mathbf{P} = (\mathbf{p_{ij}}) \in \mathbb{R}^{\mathbf{p} \times \mathbf{p}}$ is defined by

$$\mathbf{p_{ij}} = \begin{cases} \mathbf{1/deg(i)} & \text{if page } \mathbf{i} \text{ links to } \mathbf{j}, \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

The **PageRank vector** $\mathbf{r}$ is the **left** eigenvector of $\mathbf{P}$ corresponding to its dominant eigenvalue $\mathbf{1}$, that is

$$\mathbf{r} = \mathbf{P^T r}.$$

We want to compute it by the **power method**

$$\mathbf{r^{(n+1)}} = \mathbf{P^T r^{(n)}}, \quad \mathbf{n} = \mathbf{0}, \mathbf{1}, \ldots, \quad \mathbf{r^{(0)}} = \mathbf{v}.$$

Unfortunately, the power method has **convergence problems.**

Some of these problems are due to **dangling pages** (pages without outlink, that is with $\mathbf{deg(i) = 0}$) and, so, the Google matrix $\mathbf{P}$ is **not stochastic** (some of its rows are 0).

For avoiding this drawback, the Google matrix is **replaced** by another matrix $\widetilde{\mathbf{P}}$.

$$\widetilde{\mathbf{P}} = \mathbf{P} + \mathbf{d}\mathbf{w}^{\mathbf{T}}$$

where $\mathbf{w} = (\mathbf{w_1}, \ldots, \mathbf{w_p})^{\mathbf{T}}$ is a **probability vector** ($\mathbf{w} \geq \mathbf{0}$, and $(\mathbf{w}, \mathbf{e}) = \mathbf{1}$ with $\mathbf{e} = (\mathbf{1}, \ldots, \mathbf{1})^{\mathbf{T}}$),
and $\mathbf{d} = (\mathbf{d_i})$ with $\mathbf{d_i} = \mathbf{1}$ if $\mathbf{deg(i)} = \mathbf{0}$ (dangling page) and $\mathbf{d_i} = \mathbf{0}$ otherwise.

Now, $\widetilde{\mathbf{P}}$ is **stochastic**, with $\mathbf{1}$ as dominant eigenvalue, and $\mathbf{e} = (\mathbf{1}, \ldots, \mathbf{1})^{\mathbf{T}}$ as corresponding **right** eigenvector.

Another problem arises since $\widetilde{\mathbf{P}}$ is **reducible**: it can have several eigenvalues on the unit circle, thus causing convergence problems to the power method.

Moreover, $\widetilde{\mathbf{P}}$ has **several** left eigenvectors corresponding to the dominant eigenvalue $\mathbf{1}$.

Thus, $\widetilde{\mathbf{P}}$ is **replaced** by the matrix

$$\mathbf{P_c} = \mathbf{c}\widetilde{\mathbf{P}} + (\mathbf{1}-\mathbf{c})\mathbf{E}, \quad \mathbf{E} = \mathbf{e}\mathbf{v}^{\mathbf{T}}$$

with $\mathbf{c} \in [\mathbf{0}, \mathbf{1})$, and $\mathbf{v}$ a **probability vector** (*personalization* or *teleportation vector*).

$\mathbf{P_c}$ is **stochastic** and **irreducible**. It has 1 as its dominant eigenvalue with **e** as its corresponding **right** eigenvector.

The power iterations

$$\mathbf{r_c^{(n+1)} = P_c^T r_c^{(n)}}, \quad \mathbf{n = 0, 1, \ldots,} \qquad \mathbf{r_c^{(0)} = v},$$

now **converge** to the **unique** vector

$$\mathbf{r_c = P_c^T r_c}$$

which is chosen as the **PageRank vector**.

The **power method converges as $\mathbf{c^n}$**.

Google chooses $\mathbf{c = 0.85}$

n	$\mathbf{c}^n$
10	1.97e-01
20	3.88e-02
50	2.96e-04
80	2.26e-06
100	8.75e-08

For **acceleration** of the power method, see

- Kamvar, Haveliwala, Manning, Golub (2003)
- Haveliwala, Kamvar, Klein, Manning, Golub (2003)
- Brezinski, Redivo-Zaglia (2006)

EXTRAPOLATION

We want to **compute** $\mathbf{r_c}$ for a certain value of $\mathbf{c}$ (0.85 or closer to 1).

We choose **several** (smaller) values $\mathbf{c_i}$ of the parameter, and we compute the corresponding vectors $\mathbf{r_{c_i}}$.

We **interpolate** these vectors by some function of the parameter, and then we **extrapolate** the results at the desired $\mathbf{c}$.

This procedure only costs the number of iterations needed for $\max_i \mathbf{c_i}$.

Why?

We set $\widetilde{\mathbf{A}} = \widetilde{\mathbf{P}}^{\mathbf{T}}$ and $\mathbf{A_c} = \mathbf{P_c^T}$. Thus $\mathbf{r_c} = \mathbf{A_c r_c}$.

Boldi, Santini, Vigna (2005) proved that

$$\mathbf{r_c} = \mathbf{v} + \mathbf{c}(\widetilde{\mathbf{A}} - \mathbf{I}) \sum_{\mathbf{i=0}}^{\infty} \mathbf{c^i} \widetilde{\mathbf{A}}^{\mathbf{i}} \mathbf{v},$$

and that the **power method produces the partial sums of this series**, that is

$$\begin{aligned} \mathbf{r_c^{(n+1)}} &= \mathbf{v} + \mathbf{c}(\widetilde{\mathbf{A}} - \mathbf{I}) \sum_{\mathbf{i=0}}^{\mathbf{n}} \mathbf{c^i} \widetilde{\mathbf{A}}^{\mathbf{i}} \mathbf{v} \\ &= \mathbf{r_c^{(n)}} + \mathbf{c^{n+1}} (\widetilde{\mathbf{A}} - \mathbf{I}) \widetilde{\mathbf{A}}^{\mathbf{n}} \mathbf{v}, \quad \mathbf{n} = \mathbf{0}, \mathbf{1}, \ldots \end{aligned}$$

with $\mathbf{r_c^{(0)}} = \mathbf{v}$. Thus, **for any c**

$$(\widetilde{\mathbf{A}} - \mathbf{I}) \widetilde{\mathbf{A}}^{\mathbf{n}} \mathbf{v} = \frac{1}{\mathbf{c^{n+1}}} (\mathbf{r_c^{(n+1)}} - \mathbf{r_c^{(n)}}).$$

This relation shows that it is possible to **apply the power method for different values of c at a low additional cost**.

Indeed, since the vectors $(\widetilde{\mathbf{A}} - \mathbf{I})\widetilde{\mathbf{A}}^{\mathbf{n}}\mathbf{v}$ are **independent of c**, the vectors $\mathbf{r}_{\widetilde{\mathbf{c}}}^{(\mathbf{n})}$ corresponding to a **different value $\widetilde{c}$** of the parameter can be directly computed by

$$\begin{aligned} \mathbf{r}_{\widetilde{\mathbf{c}}}^{(\mathbf{0})} &= \mathbf{v} \\ \mathbf{r}_{\widetilde{\mathbf{c}}}^{(\mathbf{n+1})} &= \mathbf{r}_{\widetilde{\mathbf{c}}}^{(\mathbf{n})} + \widetilde{\mathbf{c}}^{\mathbf{n+1}} \frac{1}{\mathbf{c}^{\mathbf{n+1}}} (\mathbf{r}_{\mathbf{c}}^{(\mathbf{n+1})} - \mathbf{r}_{\mathbf{c}}^{(\mathbf{n})}), \quad \mathbf{n} = \mathbf{0}, \mathbf{1}, \ldots \end{aligned}$$

WHAT IS EXTRAPOLATION?:

Assume that the values of a function $\mathbf{f}$ are known at $\mathbf{k}$ points $\mathbf{x_i}$, that is

$$\mathbf{y_i} = \mathbf{f}(\mathbf{x_i}), \quad \mathbf{i} = \mathbf{1}, \ldots, \mathbf{k}.$$

Choose a function $\mathbf{F_k}$ belonging to some class of functions, and depending on $\mathbf{k}$ parameters: $\mathbf{F_k}(\mathbf{a_1}, \ldots, \mathbf{a_k}, \cdot)$

Compute $\mathbf{a_1^*}, \ldots, \mathbf{a_k^*}$ solution of the system of equations

$$\mathbf{F_k}(\mathbf{a_1^*}, \ldots, \mathbf{a_k^*}, \mathbf{x_i}) = \mathbf{y_i}, \qquad \mathbf{i} = \mathbf{1}, \ldots, \mathbf{k}.$$

$\mathbf{F_k}$ **interpolates** $\mathbf{f}$ at the points $\mathbf{x_i}$.

For $\mathbf{x}^* \notin [\min_\mathbf{i} \mathbf{x_i}, \max_\mathbf{i} \mathbf{x_i}]$, compute the **extrapolated** value

$$\mathbf{y}^* = \mathbf{F_k}(\mathbf{a_1^*}, \ldots, \mathbf{a_k^*}, \mathbf{x}^*).$$

EXAMPLE: ROMBERG'S METHOD:

$\mathbf{y_i} =$ result obtained by the trapezoidal rule with the step $\mathbf{h_i}$.

Set $\mathbf{x_i} = \mathbf{h_i^2}$.

$\mathbf{F_k} =$ polynomial of degree $\mathbf{k-1}$.

Extrapolate at $\mathbf{x^* = 0}$.

Why is Romberg's method working so well?

Because, by the Euler-Maclaurin formula, the results of the trapezoidal rule behave **like** a polynomial in $\mathbf{h^2}$.

EXTRAPOLATION OF THE PAGERANK VECTORS:

So, for extrapolation to work well (that is for choosing the class of functions), we have to

analyze the behavior of $\mathbf{r_c}$ with respect to $\mathbf{c}$.

Let

$\widetilde{\lambda}_1 = 1, \widetilde{\lambda}_2, \ldots, \widetilde{\lambda}_p$ the **eigenvalues** of $\widetilde{\mathbf{P}}$
$\mathbf{e}, \mathbf{x_2}, \ldots, \mathbf{x_p}$ its corresponding **right eigenvectors** and
$\widetilde{\mathbf{r}}, \mathbf{y_2}, \ldots, \mathbf{y_p}$ its corresponding **left eigenvectors**
($\widetilde{\mathbf{r}}$ is the Pagerank vector corresponding to $\mathbf{c} = \mathbf{1}$)

As proved by Serra-Capizzano (2005):

➜ if $\widetilde{\mathbf{P}}$ is **diagonalizable**

$$\mathbf{r_c} = \widetilde{\mathbf{r}} + (\mathbf{1}-\mathbf{c})\sum_{\mathbf{i=2}}^{\mathbf{p}} \frac{\alpha_\mathbf{i}}{\mathbf{1}-\mathbf{c}\widetilde{\lambda}_\mathbf{i}}\,\mathbf{y_i}, \qquad \alpha_\mathbf{i} = \mathbf{x_i^T v}$$

➜ in the **general case**

$$\mathbf{r_c} = \widetilde{\mathbf{r}} + \sum_{\mathbf{i=2}}^{\mathbf{p}} \mathbf{w_i(c)y_i} \quad \text{with}$$

$$\begin{aligned} \mathbf{w_2(c)} &= \mathbf{(1-c)}\alpha_\mathbf{2}/\mathbf{(1-c}\widetilde{\lambda}_\mathbf{2}\mathbf{)} \\ \mathbf{w_i(c)} &= [\mathbf{(1-c)}\alpha_\mathbf{i} + \mathbf{c}\,\beta_\mathbf{i}\,\mathbf{w_{i-1}(c)}]/\mathbf{(1-c}\widetilde{\lambda}_\mathbf{i}\mathbf{)}, \qquad \mathbf{i = 3,\ldots,p} \end{aligned}$$

with β_i equal to 0 or 1.

In both cases,

$\mathbf{r_c}$ is a **rational function** with a **vector numerator** of degree $\mathbf{p-1}$, and a **scalar denominator** of degree $\mathbf{p-1}$ in $\mathbf{c}$.

So, the class of functions used for **extrapolation** will be the class of **rational functions** of the same type, but of degree

$$\mathbf{k << p-1}.$$

A first account of such extrapolation procedures was given in Brezinski, Redivo–Zaglia, Serra-Capizzano, (2005).

VECTOR RATIONAL EXTRAPOLATION:

We **interpolate** the vectors $\mathbf{r_c}$ corresponding to several values of the parameter $\mathbf{c}$ by the **vector rational function**

$$\mathbf{p}(\mathbf{c}) = \frac{\mathbf{P}_k(\mathbf{c})}{Q_k(\mathbf{c})}.$$

The coefficients of $\mathbf{P}_k$ and Q_k are obtained by solving the interpolation problem

$$Q_k(\mathbf{c}_i)\mathbf{p}_i = \mathbf{P}_k(\mathbf{c}_i), \qquad i = 0, \ldots, k,$$

with $\mathbf{p}_i = \mathbf{r}_{\mathbf{c}_i}$, and the $\mathbf{c}_i$'s **distinct points** in $]0, 1[$.

$\mathbf{P}_k$ and Q_k are given by **Lagrange's interpolation formula**

$$\begin{aligned}\mathbf{P}_k(\mathbf{c}) &= \sum_{i=0}^{k} L_i(\mathbf{c})\mathbf{P}_k(\mathbf{c}_i)\\ Q_k(\mathbf{c}) &= \sum_{i=0}^{k} L_i(\mathbf{c})Q_k(\mathbf{c}_i)\end{aligned}$$

with

$$L_i(\mathbf{c}) = \prod_{\substack{j=0\\ j\neq i}}^{k} \frac{\mathbf{c}-\mathbf{c}_j}{\mathbf{c}_i-\mathbf{c}_j}, \qquad i = 0,\ldots,k.$$

Thus

$$\mathbf{P}_k(\mathbf{c}) = \sum_{i=0}^{k} L_i(\mathbf{c})Q_k(\mathbf{c}_i)\mathbf{p}_i.$$

How to compute $Q_k(\mathbf{c}_0), \ldots, Q_k(\mathbf{c}_k)$?

Assume that, for $\mathbf{c}^* \neq \mathbf{c}_i$, $i = 0, \ldots, k$, the vector $\mathbf{r}_{\mathbf{c}^*}$ is known.

Since $\mathbf{p}(\mathbf{c}) = \mathbf{P}_k(\mathbf{c})/Q_k(\mathbf{c})$, from the previous result, we will **approximate** $\mathbf{r}_{\mathbf{c}^*}$ by

$$\mathbf{p}(\mathbf{c}^*) = \sum_{i=0}^{k} L_i(\mathbf{c}^*) a_i(\mathbf{c}^*) \mathbf{p}_i,$$

with $a_i(\mathbf{c}^*) = Q_k(\mathbf{c}_i)/Q_k(\mathbf{c}^*)$.

Let $\mathbf{s}_0, \ldots, \mathbf{s}_k$ be $k+1$ **linearly independent vectors**. After taking their scalar products with the vector $\mathbf{p}(\mathbf{c}^*)$, and with the vector $\mathbf{r}_{\mathbf{c}^*}$, we will look for $a_0(\mathbf{c}^*), \ldots, a_k(\mathbf{c}^*)$ solution of the system of $k+1$ linear equations

$$\sum_{i=0}^{k} (\mathbf{p}_i, \mathbf{s}_j) L_i(\mathbf{c}^*) a_i(\mathbf{c}^*) = (\mathbf{r}_{\mathbf{c}^*}, \mathbf{s}_j), \qquad j = 0, \ldots, k.$$

Instead of a linear system in the unknowns $a_0(\mathbf{c}^*), \ldots, a_k(\mathbf{c}^*)$, we can consider it as a system in the unknowns $L_0(\mathbf{c}^*)a_0(\mathbf{c}^*), \ldots, L_k(\mathbf{c}^*)a_k(\mathbf{c}^*)$. Since the $L_i(\mathbf{c}^*)$'s are know quantities, the $a_i(\mathbf{c}^*)$ will be immediately deduced from the solution.

So, by setting $M = [\mathbf{p}_0, \ldots, \mathbf{p}_k]$, $S = [\mathbf{s}_0, \ldots, \mathbf{s}_k]$, and $\mathbf{u}(\mathbf{c}^*) = (L_0(\mathbf{c}^*)a_0(\mathbf{c}^*), \ldots, L_k(\mathbf{c}^*)a_k(\mathbf{c}^*))^T$, then the system writes

$$S^T M \mathbf{u}(\mathbf{c}^*) = S^T \mathbf{r}_{\mathbf{c}^*},$$

and it follows

$$\mathbf{p}(\mathbf{c}^*) = M\mathbf{u}(\mathbf{c}^*) = M(S^T M)^{-1} S^T \mathbf{r}_{\mathbf{c}^*}.$$

For the particular choice $\mathbf{s}_j = \mathbf{p}_j$, $j = 0, \ldots, k$, the system has a **symmetric positive definite Gram matrix** $M^T M$, and $\mathbf{p}(\mathbf{c}^*) = \sum_{i=0}^{k} L_i(\mathbf{c}^*) a_i(\mathbf{c}^*) \mathbf{p}_i$ is the **best approximation** of $\mathbf{r}_{\mathbf{c}^*}$ in $\mathrm{span}(\mathbf{p}_0, \ldots, \mathbf{p}_k)$.

In that case,

$$\mathbf{p}(\mathbf{c}^*) = M(M^T M)^{-1} M^T \mathbf{r}_{\mathbf{c}^*}$$

is the **orthogonal projection** of $\mathbf{r}_{\mathbf{c}^*}$ on $\mathrm{span}(\mathbf{p}_0, \ldots, \mathbf{p}_k)$.

Once the $a_i(\mathbf{c}^*)$'s have been obtained, the $Q_k(\mathbf{c}_i)$'s could be computed. For that, it is necessary to know the value of $Q_k(\mathbf{c}^*)$, being $a_i(\mathbf{c}^*) = Q_k(\mathbf{c}_i)/Q_k(\mathbf{c}^*)$.

Since a rational function is determined apart a multiplying factor, it does not restrict the generality to assume that the **polynomial Q_k is monic**.

So, from $Q_k(\mathbf{c}) = \sum_{i=0}^{k} L_i(\mathbf{c}) Q_k(\mathbf{c}_i)$, we see that its dominant coefficient satisfies the relation

$$1 = \sum_{i=0}^{k} \frac{Q_k(\mathbf{c}_i)}{\prod\limits_{\substack{j=0 \\ j\neq i}}^{k} (\mathbf{c}_i - \mathbf{c}_j)} = Q_k(\mathbf{c}^*) \sum_{i=0}^{k} \frac{a_i(\mathbf{c}^*)}{\prod\limits_{\substack{j=0 \\ j\neq i}}^{k} (\mathbf{c}_i - \mathbf{c}_j)},$$

which gives $Q_k(\mathbf{c}^*)$. Then,

$$Q_k(\mathbf{c}_i) = a_i(\mathbf{c}^*) Q_k(\mathbf{c}^*), \qquad i = 0, \ldots, k$$

But, it is even unnecessary to know the $Q_k(c_i)$'s and $Q_k(c^*)$.

Indeed, for an **arbitrary value** of $\mathbf{c}$, we obtain an **approximation** of $\mathbf{r_c}$ as

$$\mathbf{p}(\mathbf{c}) = \frac{\mathbf{P}_k(\mathbf{c})}{Q_k(\mathbf{c})} = \frac{\sum_{i=0}^{k} L_i(\mathbf{c}) Q_k(\mathbf{c}_i) \mathbf{p}_i}{\sum_{i=0}^{k} L_i(\mathbf{c}) Q_k(\mathbf{c}_i)}.$$

Dividing the numerator and the denominator by $Q_k(\mathbf{c}^*)$ finally leads to the **extrapolation formula**

$$\mathbf{p}(\mathbf{c}) = \frac{\sum_{i=0}^{k} L_i(c) a_i(c^*) \mathbf{p}_i}{\sum_{i=0}^{k} L_i(c) a_i(c^*)}.$$

Vector rational extrapolation procedure

1. Choose $k+2$ distinct values of $\mathbf{c}$: $\mathbf{c}_0, \ldots, \mathbf{c}_k$ and $\mathbf{c}^*$.
2. Compute $\mathbf{p}_i = \mathbf{r}_{\mathbf{c}_i}$ for $i = 0, \ldots, k$, and $\mathbf{r}_{\mathbf{c}^*}$ (low cost formula).
3. Choose $k+1$ linearly independent vectors $\mathbf{s}_0, \ldots, \mathbf{s}_k$, or take $\mathbf{s}_i = \mathbf{p}_i$ for $i = 0, \ldots, k$.
4. Compute $a_0(\mathbf{c}^*), \ldots, a_k(\mathbf{c}^*)$ by solving the system

$$\sum_{i=0}^{k} (\mathbf{p}_i, \mathbf{s}_j) L_i(\mathbf{c}^*) a_i(\mathbf{c}^*) = (\mathbf{r}_{\mathbf{c}^*}, \mathbf{s}_j), \qquad j = 0, \ldots, k,$$

or

$$S^T M \mathbf{u}(\mathbf{c}^*) = S^T \mathbf{r}_{\mathbf{c}^*}.$$

5. Compute an approximation of $\mathbf{r}_{\mathbf{c}}$ by

$$\mathbf{p}(\mathbf{c}) = \frac{\sum_{i=0}^{k} L_i(\mathbf{c}) a_i(\mathbf{c}^*) \mathbf{p}_i}{\sum_{i=0}^{k} L_i(\mathbf{c}) a_i(\mathbf{c}^*)}.$$

A SIMPLER VECTOR RATIONAL EXTRAPOLATION:

Let us construct a vector rational extrapolation method by **truncating** the rational expression of $\mathbf{r_c}$ given by Serra-Capizzano, **after two terms**.

So, we consider an extrapolation function of the form

$$\mathbf{p}(\mathbf{c}) = \mathbf{y} + (1 - \mathbf{c})\frac{1}{1 - \mathbf{c}\lambda}\,\mathbf{z}.$$

These two unknown vectors and the unknown scalar will be computed by an **interpolation** procedure needing only 3 values of $\mathbf{c}$.

We will compute first the scalar λ, after the vector $\mathbf{z}$ and, finally, the vector $\mathbf{y}$ obtained as $\mathbf{p}(1)$.

As above, set $\mathbf{p}_i = \mathbf{r}_{\mathbf{c}_i}$.
We consider the interpolation condition

$$\mathbf{p}_i = \mathbf{y} + \frac{1 - \mathbf{c}_i}{1 - \mathbf{c}_i \lambda} \mathbf{z}.$$

The difference $\mathbf{p}_i - \mathbf{p}_j$ eliminates $\mathbf{y}$, and we have

$$\mathbf{p}_i - \mathbf{p}_j = \frac{(\mathbf{c}_j - \mathbf{c}_i)(1 - \lambda)}{(1 - \mathbf{c}_i \lambda)(1 - \mathbf{c}_j \lambda)} \mathbf{z}.$$

We now need to compute the scalar λ and the vector $\mathbf{z}$.

Let $\mathbf{u}$ be a vector so that the scalar products $(\mathbf{p}_i - \mathbf{p}_j, \mathbf{u})$ and $(\mathbf{p}_k - \mathbf{p}_j, \mathbf{u})$ are different from zero.

We set

$$r_{ijk} = \frac{(\mathbf{p}_i - \mathbf{p}_j, \mathbf{u})}{(\mathbf{p}_k - \mathbf{p}_j, \mathbf{u})} = \frac{\mathbf{c}_j - \mathbf{c}_i}{\mathbf{c}_j - \mathbf{c}_k} \frac{1 - \mathbf{c}_k \lambda}{1 - \mathbf{c}_i \lambda},$$

which gives

$$\lambda = \frac{r_{ijk}(\mathbf{c}_j - \mathbf{c}_k) - (\mathbf{c}_j - \mathbf{c}_i)}{\mathbf{c}_i\, r_{ijk}(\mathbf{c}_j - \mathbf{c}_k) - \mathbf{c}_k(\mathbf{c}_j - \mathbf{c}_i)}.$$

Then $\mathbf{z}$ follows

$$\mathbf{z} = \frac{(1 - \mathbf{c}_i\lambda)(1 - \mathbf{c}_j\,\lambda)}{(\mathbf{c}_j - \mathbf{c}_i)(1 - \lambda)}(\mathbf{p}_i - \mathbf{p}_j).$$

Finally, $\mathbf{y}$ is given by

$$\mathbf{y} = \mathbf{p}(1) = \mathbf{p}_i - \frac{1 - \mathbf{c}_i}{1 - \mathbf{c}_i \lambda}\mathbf{z}.$$

Thus, from the expressions for λ, $\mathbf{z}$ and $\mathbf{y}$ we obtain an **approximation** of the vector $\mathbf{r_c}$, for a chosen $\mathbf{c}$, given by

$$\mathbf{r_c} \simeq \mathbf{p}(\mathbf{c}) = \mathbf{y} + (1 - \mathbf{c})\frac{1}{1 - \mathbf{c}\lambda}\mathbf{z}.$$

A MINIMIZATION PROCEDURE:

We consider an approximation $\mathbf{p(c)}$ of $\mathbf{r_c}$ of the form

$$\mathbf{p(c)} = (1-\alpha)\mathbf{p}_0 + \alpha\mathbf{p}_1 = \mathbf{p}_0 + \alpha(\mathbf{p}_1 - \mathbf{p}_0),$$

where the parameter α is chosen so that the **euclidean norm** of $P_{\mathbf{c}}^T\mathbf{p(c)} - \mathbf{p(c)}$ (a vector which could be considered as a residual since $P_{\mathbf{c}}^T\mathbf{r_c} - \mathbf{r_c}$) **is minimum**.

It holds

$$\alpha = -\frac{(P_{\mathbf{c}}^T(\mathbf{p}_1 - \mathbf{p}_0) - (\mathbf{p}_1 - \mathbf{p}_0), P_c^T\mathbf{p}_0 - \mathbf{p}_0)}{\|P_{\mathbf{c}}^T(\mathbf{p}_1 - \mathbf{p}_0) - (\mathbf{p}_1 - \mathbf{p}_0)\|^2}.$$

This procedure needs of only 2 values of $\mathbf{c}$. Obviously it could be extended to a more general form of minimization where

$$\mathbf{p(c)} = \alpha_0\mathbf{p}_0 + \cdots + \alpha_k\mathbf{p}_k$$

with $\alpha_0 + \cdots + \alpha_k = 1$.

NUMERICAL EXPERIMENTS

$P = (p_{ij})$ is randomly constructed. Dimension p.

First we select a random integer q between 1 and $p/10$.

Then, we generate a random integer vector $\mathbf{m}$ of dimension p with components between 1 and q.

Each **row** i of our matrix P will contain, at most, $\mathbf{m}(i)$ nonzero elements.

Then, we randomly choose, for each i, an integer vector of dimension $\mathbf{m}(i)$, with components between 1 and p, and we eliminate its identical components and those equal to i.

The length of the reduced vector is $\deg(i) \leq \mathbf{m}(i)$, and its components give the **indexes** j of the **columns** such that $p_{ij} = 1/\deg(i)$, all others elements being set to zero.

Finally, among all rows, we randomly set to zero $p/5$ of them, corresponding to the dangling nodes.

Such matrices P (and the corresponding matrices $\widetilde{P}$ and $P_{\mathbf{c}}$) have the **same properties** as those coming out from the web.

A very important point to mention, is that we are not interested in the exact values of the components of the real and extrapolated PageRank vectors, but in their relative values, that is the rank of each of them compared with the other components.

The values and the ranks can be quite sensitive:

- stability of PageRank algorithm (Lempel, Moran, 2005)
- rank-stability (Borodin et al., 2005)
- detailed explanations (Langville, Meyer, 2006)

Example (Ipsen, ANAW 2006, Pisa) :

$$\mathbf{r_c} = (\; 0.23 \quad 0.24 \quad 0.26 \quad 0.27 \;)^T$$

$$\text{rank}(\mathbf{r_c}) = \quad 4 \quad 3 \quad 2 \quad 1$$

$$\mathbf{r_1(c)} = (\; 0.27 \quad 0.26 \quad 0.24 \quad 0.25 \;)^T$$

$$\text{rank}(\mathbf{r_1(c)}) = \quad 1 \quad 2 \quad 4 \quad 3$$

$\|\mathbf{r_c} - \mathbf{r_1(c)}\|_\infty = 0.04$ (**small error**, but **incorrect ranking**)

$$\mathbf{r_2(c)} = (\; 0 \quad 0.001 \quad 0.002 \quad 0.997 \;)^T$$

$$\text{rank}(\mathbf{r_2(c)}) = \quad 4 \quad 3 \quad 2 \quad 1$$

$\|\mathbf{r_c} - \mathbf{r_2(c)}\|_\infty = 0.727$ (**bigger error**, but **correct ranking**)

Notations for the methods

In all the examples, we choose 9 different values for $\mathbf{c}$:
$\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_7$ and $\mathbf{c}^*$

VREM n $\longrightarrow$ Vector rational extrapolation with
$\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_{\mathbf{n}-2}$ and $\mathbf{c}^*$.

SVREM 3 $\longrightarrow$ Simpler vector rational extrapolation with
$\mathbf{c}_5, \mathbf{c}_6, \mathbf{c}_7$ (that is $\mathbf{p}_5, \mathbf{p}_6, \mathbf{p}_7$), and $\mathbf{u} = \mathbf{p}_7 - \mathbf{p}_5$.

VMP 2 $\longrightarrow$ Minimization procedure with
$\mathbf{c}_5, \mathbf{c}_7$ (that is $\mathbf{p}_5$ and $\mathbf{p}_7$).

All results were obtained with $\mathbf{w} = \mathbf{v}$.

Notations for the results

nch $\longrightarrow$ total number of changes in the ranking
between the Pagerank vector $\mathbf{r_c}$ and
the extrapolated vector $\mathbf{p(c)}$.

ich $\longrightarrow$ rank of the first change occurred
after sorting by descending values $\mathbf{r_c}$ and $\mathbf{p(c)}$.

$\mathbf{d}_{\max}$ $\longrightarrow$ maximum displacement of a page.
A positive value of $d_{\max}$ means that the
corresponding page went up in the list,
and that it went down if it is negative.

$ix_{\max}$, $iy_{\max}$ $\longrightarrow$ The ranks of the page corresponding to $\mathbf{d}_{\max}$
$ix_{\max}$ in the sorted $\mathbf{r_c}$
$iy_{\max}$ in the sorted $\mathbf{p(c)}$.

First example

$p = 5000, nnz = 942806.$

Google parameter $\mathbf{c} = 0.85$.

8 iterations with power method for a precision of 10^{-8}.

The highest and the smallest components of the PageRank vector were $3.84636884 \cdot 10^{-4}$ and $1.48826460 \cdot 10^{-4}$, respectively, thus meaning that, when p is large, many components can differ only in the last digits.

#	Method	$\|\mathbf{r}_c - \mathbf{p}\|_\infty$	$\|\mathbf{r}_c - \mathbf{p}\|_1/p$	nch	ich	$\mathbf{d}_{\max}$	$ix_{\max}$	$iy_{\max}$
1	VREM 4	2.43e-6	2.57e-8	4417	18	-47	1553	1600
2	VREM 5	5.13e-8	3.32e-9	1667	29	6	2651	2645
3	VREM 6	6.03e-8	2.34e-9	1254	190	-4	2358	2362
4	**VREM 7**	2.77e-8	1.24e-9	**689**	**190**	**2**	890	888
5	VREM 8	3.04e-8	1.89e-9	1029	190	-4	2358	2362
6	VREM 9	2.87e-8	1.74e-9	939	190	4	2765	2761
7	VREM 4	6.86e-7	1.04e-8	3479	18	12	2691	2679
8	SVREM 3	1.17e-5	9.68e-8	4863	1	-234	3529	3763
9	VMP 2	1.20e-5	7.88e-8	4865	1	-236	3529	3765

$p = 5000$, $\mathbf{c} = 0.85$ (8 iterations)

$\mathbf{c}_i = 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45,$ $\mathbf{c}^* = 0.5$ (5 iterations)

Quality of the results

It seems that the most two important parameters to consider are $\mathbf{d}_{\max}$ and **ich**.

$\mathbf{d}_{\max}$ indicates the **size of the largest change in the ranking**.
The **smallest** $\mathbf{d}_{\max}$, the **better the ranking**.
So, a criterion of good quality is to have a small value of $\mathbf{d}_{\max}$.
But $\mathbf{d}_{\max}$ can be **large** if **ich** is also **large**.

In fact, **ich** indicates the **location of the first change** in the ranking. So, a **correct ranking** has been obtained for the **ich-1 first components of the extrapolated vector**.

It is **not so important** to have many changes **(ich large)** in the ranking if they are small, that is if $\mathbf{d}_{\max}$ **is small**.

Interchanging $\mathbf{c}_3 = 0.25$ and $\mathbf{c}^* = 0.5$ does not change much the results. Best method for both tests seems to be **VREM 7**. Here **VREM 5** give comparable results.

#	Method	$\|\mathbf{r}_c - \mathbf{p}\|_\infty$	$\|\mathbf{r}_c - \mathbf{p}\|_{1/p}$	**nch**	**ich**	$\mathbf{d}_{\max}$	$ix_{\max}$	$iy_{\max}$
1	VREM 4	2.43e-6	2.57e-8	4420	18	-47	1553	1600
2	**VREM 5**	4.31e-8	2.01e-9	**1102**	**190**	**-4**	2358	2362
3	VREM 6	3.07e-8	1.86e-9	1010	190	-4	2358	2362
4	**VREM 7**	2.50e-8	1.54e-9	**827**	**190**	**4**	2765	2761
5	VREM 8	3.10e-8	1.90e-9	1033	190	-4	2358	2362
6	VREM 9	9.59e-7	2.71e-8	4520	10	-57	2710	2767
7	VREM 4	3.53e-5	2.25e-6	4808	10	127	2362	2235
8	SVREM 3	1.17e-5	9.68e-8	4863	1	-234	3529	3763
9	VMP 2	1.20e-5	7.88e-8	4865	1	-236	3529	3765

Same matrix with $\mathbf{c}_i$'s closer to 0.85:

$\mathbf{c}_i = 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6,$ 0.65 (6 iterations), $\mathbf{c}^* = 0.25$

$\mathbf{c} = 0.85$ (8 iterations)

#	Method	$\|\mathbf{r}_c - \mathbf{p}\|_\infty$	$\|\mathbf{r}_c - \mathbf{p}\|_1/p$	nch	ich	$\mathbf{d}_{\max}$	$ix_{\max}$	$iy_{\max}$
1	VREM 4	9.30e-7	1.31e-8	3788	18	-16	2461	2477
2	VREM 5	2.01e-8	1.17e-9	635	207	3	2765	2762
3	VREM 6	1.14e-8	6.92e-10	385	251	2	936	934
4	**VREM 7**	2.65e-9	1.29e-10	**66**	**272**	**-1**	272	273
5	VREM 8	3.16e-9	2.02e-10	114	272	-1	272	273
6	VREM 9	2.07e-9	1.25e-10	66	272	-1	272	273
7	VREM 4	3.52e-5	2.25e-6	4810	10	128	2362	2234
8	SVREM 3	4.02e-6	4.48e-8	4673	14	-101	3529	3630
9	VMP 2	4.15e-6	4.07e-8	4683	14	-102	3529	3631

Extrapolation for $\mathbf{r_c}$ with $\mathbf{c} = 0.99$ (13 iterations) and
$\mathbf{c}_i = 0.55, 0.6, 0.65$ (6 iterations), $\mathbf{c}^* = 0.25$
gives

#	Method	$\|\mathbf{r}_c - \mathbf{p}\|_\infty$	$\|\mathbf{r}_c - \mathbf{p}\|_1/p$	**nch**	**ich**	$\mathbf{d}_{\max}$	$ix_{\max}$	$iy_{\max}$
7	VREM 4	3.52e-5	2.25e-6	4810	10	128	2362	2234
8	**SVREM 3**	3.08e-5	2.30e-6	**4349**	**29**	**-40**	3529	3569
9	**VMP 2**	3.03e-5	2.27e-6	**4359**	**29**	**-40**	3529	3569

Example 2: Stanford web matrix

$p = 281903, nnz = 2312497,$ $\mathbf{c} = 0.85$ (91 iterations)

$\mathbf{c}_i = 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45,$ $\mathbf{c}^* = 0.5$ (22 iterations)

#	Method	$\|\mathbf{r}_c - \mathbf{p}\|_\infty$	$\|\mathbf{r}_c - \mathbf{p}\|_1/p$	nch	ich	$\mathbf{d}_{\max}$	$ix_{\max}$	$iy_{\max}$
1	VREM 4	1.22e-3	6.26e-7	261573	4	-162408	26841	189249
2	VREM 5	1.78e-3	1.40e-7	261445	4	-105526	19635	125161
3	VREM 6	7.67e-4	1.02e-7	261208	4	-89744	52409	142153
4	VREM 7	4.52e-4	7.50e-8	260291	4	-44139	32553	76692
5	VREM 8	3.00e-4	5.25e-8	260629	4	-52413	116455	168868
6	**VREM 9**	2.57e-4	6.93e-8	**281652**	**11**	**-219944**	61958	281902
7	VREM 4	2.00e-3	6.74e-7	261353	4	-80743	51569	132312
8	SVREM 3	1.59e-3	8.74e-7	261573	1	-160026	19635	179661
9	VMP 2	1.51e-3	9.27e-7	261574	1	-160620	19635	180255

Then, we will consider extrapolation with larger values.

$p = 281903, nnz = 2312497,\ \mathbf{c} = 0.85$ (91 iterations)

$\mathbf{c}_i = 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65$ (39 iterations), $\mathbf{c}^* = 0.25$

#	Method	$\|\mathbf{r}_c - \mathbf{p}\|_\infty$	$\|\mathbf{r}_c - \mathbf{p}\|_1/p$	nch	ich	$\mathbf{d}_{\max}$	$ix_{\max}$	$iy_{\max}$
1	VREM 4	1.05e-3	4.15e-7	261425	4	-104574	26841	131415
2	VREM 5	5.26e-4	9.61e-8	261240	4	-54793	32553	87346
3	VREM 6	4.02e-4	5.95e-8	260085	7	-37547	32553	70100
4	VREM 7	9.55e-5	1.45e-8	258487	14	-23600	32553	56153
5	**VREM 8**	3.32e-5	7.33e-9	**257896**	**29**	**-20639**	32553	53192
6	VREM 9	1.03e-5	2.98e-9	254360	14	-13364	38289	51653
7	VREM 4	7.72e-4	2.76e-7	260586	4	-37167	32553	69720
8	SVREM 3	1.59e-3	2.95e-7	261569	1	-79828	51569	131397
9	VMP 2	1.01e-3	4.61e-7	261576	1	-81710	51569	133279

Example 3: A small matrix

$p = 1000, nnz = 18729,$ $\mathbf{c} = 0.85$ (12 iterations)

$\mathbf{c}_i = 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45,$ $\mathbf{c}^* = 0.5$ (9 iterations)

#	Method	$\|\mathbf{r}_c - \mathbf{p}\|_\infty$	$\|\mathbf{r}_c - \mathbf{p}\|_1/p$	nch	ich	$\mathbf{d}_{\max}$	$ix_{\max}$	$iy_{\max}$
1	VREM 4	5.14e-5	1.91e-6	913	9	-51	348	399
2	VREM 5	8.29e-6	5.44e-7	603	12	9	398	389
3	VREM 6	1.39e-6	1.15e-7	175	13	4	486	482
4	VREM 7	1.23e-6	1.03e-7	152	62	4	486	482
5	**VREM 8**	5.38e-7	4.60e-8	**81**	**127**	**-2**	395	397
6	**VREM 9**	2.66e-7	2.35e-8	**46**	**127**	**2**	145	143
7	VREM 4	5.42e-6	4.46e-7	615	15	-7	467	474
8	SVREM 3	1.17e-4	3.63e-6	969	8	101	425	324
9	VMP 2	1.35e-4	3.77e-6	966	8	101	425	324

REFERENCE:

C. Brezinski, M. Redivo-Zaglia, *Rational extrapolation for the PageRank vector*, **submitted**

FUTURE WORKS:

➜ Study **other extrapolation algorithms**:

- ϱ-algorithm
- restart

➜ **Other applications:**

- Networks

- M.E.J. Newman, The structure and function of complex networks, SIAM Rev., 45 (2003) 167–256.
- E. Ravasz, A.-L. Barabási, Hierarchical organization in complex networks, Phys. Rev., E 67, 026112 (2003).
- R. Milo, S. Itzkovitz, N. Kashtan, R. Levitt, S. Shen-Orr, I. Ayzenshtat, M. Sheffer, U. Alon, Superfamilies of designed and evolved networks, Science, 303 (2004) 1538–1542.

- Small (but **reducible**) matrices.

- Kerner, 2006: agglomeration of viruses (matrices of dimensions 10 or 20!)