

Matrix Scaling and Applications in Electoral Systems

M. Zachariasen

University of Copenhagen

DIKU, December 2006

Joint work with...

Sebastian Maier (University of Augsburg)

Friedrich Pukelsheim (University of Augsburg)

Günter Rote (Freie Universität Berlin)

Petur Zachariasen (University of the Faroe Islands)

Outline

- Electoral systems
 - Basic principles
 - Types of electoral systems
 - Proportional representation systems
- Vector scaling
 - Problem definition
 - Quota-based methods
 - Divisor-based methods
 - Rounding and signpost functions
- Matrix scaling
 - Problem definition
 - Iterative proportional fitting
 - Divisor-based methods
 - Alternating scaling
 - Tie-and-transfer
 - Formulation as optimization problem

Outline

- Electoral systems
 - Basic principles
 - Types of electoral systems
 - Proportional representation systems
- Vector scaling
 - Problem definition
 - Quota-based methods
 - Divisor-based methods
 - Rounding and signpost functions
- Matrix scaling
 - Problem definition
 - Iterative proportional fitting
 - Divisor-based methods
 - Alternating scaling
 - Tie-and-transfer
 - Formulation as optimization problem

Outline

- Electoral systems
 - Basic principles
 - Types of electoral systems
 - Proportional representation systems
- Vector scaling
 - Problem definition
 - Quota-based methods
 - Divisor-based methods
 - Rounding and signpost functions
- Matrix scaling
 - Problem definition
 - Iterative proportional fitting
 - Divisor-based methods
 - Alternating scaling
 - Tie-and-transfer
 - Formulation as optimization problem

What is an electoral systems?

An **electoral system** consists of several parts:

- Size of parliament, division of electoral region into districts, **translation of vote counts into seats**, electoral threshold(s)
- Electoral rights, parties, party coalitions, method of election
- Administrative rules related to undertaking elections

Importance of electoral systems

An electoral system is an important **political tool** in a democracy: It heavily influences the way politicians work and the way the population reacts.

Central parts of an electoral system are specified in the **constitution**.

Changes to electoral systems happen slowly and seldomly.

Basic principles

Basic principles for good electoral systems may be summarized in the following conflicting objectives (Gallagher, 2005):

- 1 Accuracy of representation of voters' preferences
- 2 Socio-demographic representation in parliament (age, sex, religion, district, etc.)
- 3 Personal accountability of members to constituents
- 4 Maximization of participation opportunities for voters
- 5 Cohesive and disciplined parties
- 6 Stable effective government
- 7 Identifiability of government options
- 8 Opportunity for voters to eject governments from office

Types of electoral systems

- 1 Proportional representation systems [▶ more](#)
 - Multi-member districts
 - Seat numbers (more or less) proportional to number of votes
 - Electoral thresholds keep small parties out
- 2 Plurality voting systems [▶ more](#)
 - Single-member districts
 - Winner takes it all
 - Outcome depends on the geographic partition into districts
- 3 Mixed systems [▶ more](#)

Proportional representation system

The basic principle in a proportional representation system is that

$$\frac{\text{number of votes for a party}}{\text{total number of votes}} \approx \frac{\text{number of seats given to a party}}{\text{total number of seats}}$$

should be fulfilled for all parties.

Multiple districts make it difficult to maintain this equality both at the district level and at the regional level.

Vector scaling

Given: n -vector $\mathbf{p} = (p_i)$ of non-negative numbers, and a positive number h . Let $p = \sum_i p_i$.

Compute: Scaled vector $\mathbf{x} = (x_i)$ such that

$$\sum_i x_i = h$$

$$\frac{x_i}{h} \approx \frac{p_i}{p}, \quad \forall i$$

Trivial if x_i is allowed to be a fractional number: $x_i = (p_i/p)h$.

The challenge appears when x_i must be an **integer** as in electoral systems applications.

Integer vector scaling / Proportional apportionment

Given: n -vector $\mathbf{p} = (p_i)$ of non-negative integers (vote numbers), and a positive integer h (house size). Let $p = \sum_i p_i$.

Compute: Scaled **integer** vector (seat numbers) $\mathbf{x} = (x_i)$ such that

$$\sum_i x_i = h$$

$$\frac{x_i}{h} \approx \frac{p_i}{p}, \quad \forall i$$

In almost all cases it is **not** possible to find a solution that gives perfect proportionality, so we must balance the error evenly.

Quota-based methods

Main idea: Compute x_i based on the **quota** $q_i = (p_i/p)h$ which is the ideal (fractional) value that gives perfect proportionality.

Let $r_i = q_i - \lfloor q_i \rfloor$ be the fractional remainder of q_i .

Algorithm:

- 1 Set $x_i = \lfloor q_i \rfloor$ for $i = 1, \dots, n$.
- 2 Let $R = h - \sum_i x_i$ be the number of seats that remain to be allocated.
- 3 If $R > 0$, then allocate **one more seat** to the R parties that have the **largest** fractional remainders r_i .

Exercise 1/3: Show that the above algorithm can be implemented to run in $\Theta(n)$ time.

Divisor-based methods

Based on a rounding function $[\cdot]$ that rounds a fractional number q to either $\lfloor q \rfloor$ or $\lceil q \rceil$. Could for example round to the nearest integer.

Main idea: Compute a **multiplier** $\lambda > 0$ such that

$$\sum_i [\lambda p_i] = h$$

and set $x_i = [\lambda p_i]$.

That is: Scale the elements of vector $\mathbf{p} = (p_i)$ with some value $\lambda > 0$, such that when the scaled values are rounded the sum is h .

Rounding and signpost functions

Define rounding function $[\cdot]$ based on a **signpost function** $d(z)$, that maps an integer z to a fractional number in the interval $[z, z + 1]$:

- If $q < d(\lfloor q \rfloor)$, then $[q] = \lfloor q \rfloor$
- If $q > d(\lfloor q \rfloor)$, then $[q] = \lfloor q \rfloor + 1$.
- If $q = d(\lfloor q \rfloor)$, then $[q]$ can either be $\lfloor q \rfloor$ or $\lfloor q \rfloor + 1$.

Note that we have $d(z - 1) \leq q \leq d(z)$ when $z = [q]$.

Classical rounding functions:

Adams

$$d(z) = z$$

Sainte-Laguë/Webster

$$d(z) = z + 0.5$$

d'Hondt/Jefferson

$$d(z) = z + 1$$

Divisor-based methods: Algorithms

Classical approach: Pick the h largest fractions among

$$\begin{aligned} & p_1/d(0), p_1/d(1), p_1/d(2), \dots \\ & p_2/d(0), p_2/d(1), p_2/d(2), \dots \\ & \vdots \\ & p_n/d(0), p_n/d(1), p_n/d(2), \dots \end{aligned}$$

A trivial implementation of this method takes $O(hn)$ time. It is possible to solve the problem in $\Theta(n)$ time [Rote & Vogel, 1993].

Exercise 2/3: Argue that the classical approach for computing a divisor-based apportionment gives the correct result.

Comparison of quota- and divisor-based methods

Quota-methods:

- ✓ Scaled integer values “close” (in absolute terms) to ideal fractional values
- ✓ No quota breaches
- ✓ Does not fulfill some elementary proportionality principles, and gives rise to several so-called **paradoxes**

Divisor-methods:

- ✓ Scaled integer values “close” (in relative terms) to ideal fractional values
- ✓ Quota breaches possible
- ✓ Fulfill a number of elementary proportionality principles, and do **not** give rise to any of the paradoxes of quota-methods

Matrix scaling

Given: $n \times m$ matrix $\mathbf{P} = (p_{ij})$ of non-negative numbers, non-negative row-sum requirements $\mathbf{r} = (r_i)$ and non-negative column-sum requirements $\mathbf{c} = (c_j)$, where $\sum_i r_i = \sum_j c_j = h$.

Compute: Scaled $n \times m$ matrix \mathbf{X} , such that

- $\sum_j x_{ij} = x_{i*} = r_i, \quad \forall i$
- $\sum_i x_{ij} = x_{*j} = c_j, \quad \forall j$
- \mathbf{X} is “proportional” to \mathbf{P}

Ideally we would like both row-wise, column-wise and global proportionality. Non-trivial problem even without integer-constraints on \mathbf{X} .

Iterative proportional fitting for matrix scaling

Compute **row multipliers** λ_i and **column multipliers** μ_j such that when we set $x_{ij} = \lambda_i p_{ij} \mu_j$ then the row- and column-requirements are fulfilled.

Can be shown that if the problem is feasible, then the solution is **unique**. Is usually solved **approximately**, since rational solutions to the multipliers do not necessarily exist.

Iterative proportional fitting algorithm: Pick any starting values for λ_i and μ_j . Iterate as follows until the row- and column-sums match well enough.

Even iterations: Choose λ_i such that $\sum_j \lambda_i p_{ij} \mu_j = \lambda_i \sum_j x_{ij} \mu_j = r_i$.

Odd iterations: Choose μ_j such that $\sum_i \lambda_i p_{ij} \mu_j = \mu_j \sum_i \lambda_i x_{ij} = c_j$.

Integer matrix scaling / Biproportional apportionment

Given: Matrix \mathbf{P} , and vectors \mathbf{r} and \mathbf{c} as above. \mathbf{P} contains vote counts while \mathbf{r} and \mathbf{c} specify **seat requirements for parties and districts**, respectively.

Compute: Scaled **integer** matrix (seat numbers) \mathbf{X} , such that

- $\sum_j x_{ij} = x_{i*} = r_i, \quad \forall i$
- $\sum_i x_{ij} = x_{*j} = c_j, \quad \forall j$
- \mathbf{X} is “proportional” to \mathbf{P}

Exercise 3/3: Show that the feasibility of this problem can be checked by solving a maximum flow problem. Hint: Consider rows and columns as vertices, and add a source and a sink. In addition to the row- and column-sum requirements we only demand that if $p_{ij} = 0$, then $x_{ij} = 0$.

Divisor-based methods

Compute **row multipliers** λ_i and **column multipliers** μ_j such that when we set

$$x_{ij} = \lfloor \lambda_i p_{ij} \mu_j \rfloor$$

then the row- and column-requirements are fulfilled.

Equivalent to computing multipliers λ_i and μ_j , and integer seat numbers x_{ij} such that

$$d(x_{ij} - 1) \leq \lambda_i p_{ij} \mu_j \leq d(x_{ij}), \quad \forall (i, j)$$

General algorithmic framework

General approach:

Iteratively adjust the multipliers until the row- and column-sum requirements are fulfilled.

For any set of multipliers and corresponding solution \mathbf{X} , the **error function**

$$\Delta(\mathbf{X}) = \frac{1}{2} \sum_i |x_{i*} - r_i| + \frac{1}{2} \sum_j |x_{*j} - c_j|$$

is iteratively minimized. When $\Delta(\mathbf{X}) = 0$, a valid set of multipliers has been found, and the algorithm terminates.

Alternating scaling

Even iterations: Solve the vector apportionment problem with vote counts $(p_{i1}\mu_1, p_{i2}\mu_2, \dots, p_{im}\mu_m)$ and house size r_i for each row i .

Odd iterations: Solve the vector apportionment problem with vote counts $(\lambda_1 p_{1j}, \lambda_2 p_{2j}, \dots, \lambda_n p_{nj})$ and house size c_j is for each column j .

The running time per iteration is $O(nm)$, since we can solve the vector apportionment problem in linear time.

The algorithm does not necessarily converge if there are ties in the rounding function (but this happens very seldomly in practice).

Tie-and-transfer

Systematically updates the multipliers such that the error function reduces by exactly 1.

Assume in the following that the columns continuously add up correctly: $x_{*j} = c_j, \forall j$. Can be achieved by setting $\lambda_i = 1$ for all rows, and solving the vector apportionment problem for each column.

Main steps of tie-and-transfer iteration:

- 1 Create a chain of **ties** by updating the multipliers — but without updating \mathbf{X} .
- 2 Update \mathbf{X} in such a way that the error function is decreased by 1 without changing the multipliers (a so-called **transfer**).

Tie-and-transfer: Creating ties

Search for path in a bipartite graph $G = (I \cup J, E)$ with a vertex set consisting of the rows I and columns J of \mathbf{X} .

Edge set E consists of:

- 1 For each $i \in I$ and $j \in J$ where $p_{ij} > 0$ and $\lambda_i p_{ij} \mu_j = d(x_{ij})$ we have an edge $(i, j) \in E$. This means that x_{ij} may be **rounded up** without changing the multipliers.
- 2 For each $i \in I$ and $j \in J$ where $p_{ij} > 0$ and $\lambda_i p_{ij} \mu_j = d(x_{ij} - 1)$ we have an edge $(j, i) \in E$. This means that x_{ij} may be **rounded down** without changing the multipliers.

By systematically updating the multipliers, we can create a path from some vertex $i \in I$ where $x_{i*} < r_i$ to some vertex $i' \in I$ where $x_{i'*} > r_{i'}$.

Tie-and-transfer: Performing a transfer

A path p in G from $i \in I$ to $i' \in I$ corresponds to alternately rounding x_{ij} up and down along the edges of p .

Net effect of this so-called transfer:

- increases the sum of row i by 1
- decreases the sum of row i' by 1
- does not change the sum of other rows or the sum of any column

Running time: $O(nm(n+m))$, since we need to make at most $n+m$ breadth-first searches in a graph with $n+m$ vertices and at most nm edges.

Formulation as optimization problem

May formulate the integer matrix scaling problem as the following optimization problem [Gaffke & Pukelsheim, 2006]:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^n \sum_{j=1}^m \sum_{z=0}^{x_{ij}-1} \log \frac{d(z)}{p_{ij}} \\ \text{subject to} & \sum_{j=1}^m x_{ij} = r_i, \quad \forall i \\ & \sum_{i=1}^n x_{ij} = c_j, \quad \forall j \\ & x_{ij} \geq 0, \quad x_{ij} \text{ integer} \\ & x_{ij} = 0 \quad \text{if } p_{ij} = 0 \end{array}$$