Simplex method

CE 377K

March 31, 2015

REVIEW

Linear programming standard form

Extreme points

Basic and nonbasic variables

OUTLINE

- Application of linear programming to bridge maintenance
- 20-minute matrix algebra
- Simplex method
- Example

BRIDGE MAINTENANCE

National Bridge Investment Analysis System (NBIAS)

Software used by Federal Highway Administration to support development of national bridge investment strategies.

Bridges consist of multiple elements (deck, girder, etc.), each element can take one of five states.

Bridges are classified into strata (weather, traffic, functional class, etc.)

A deterioration model predicts degradation of each element based on the stratum

Various repair actions can be undertaken to each element, with associated costs.

The benefit of repair is the reduction in user cost due to accident/detour cost, and the increase in element "value"

NBIAS solves a linear program which minimizes cost in the long run.

Application in Texas, with roughly 50,000 bridges and a \$200 million budget...

FEDERAL HIGHWAY ADMINISTRATION 08/13/2003
NATIONAL BRIDGE INVESTMENT ANALYSIS SYSTEM 11:25:00 PM

BRIDGE NETWORK PERFORMANCE ANALYSIS REPORT Page 1 of 4

Report: FC 01 Benefit-Cost Analysis.RP21 (Performance by bridge category and year for a selected indicator and fixed B/C cutoff)

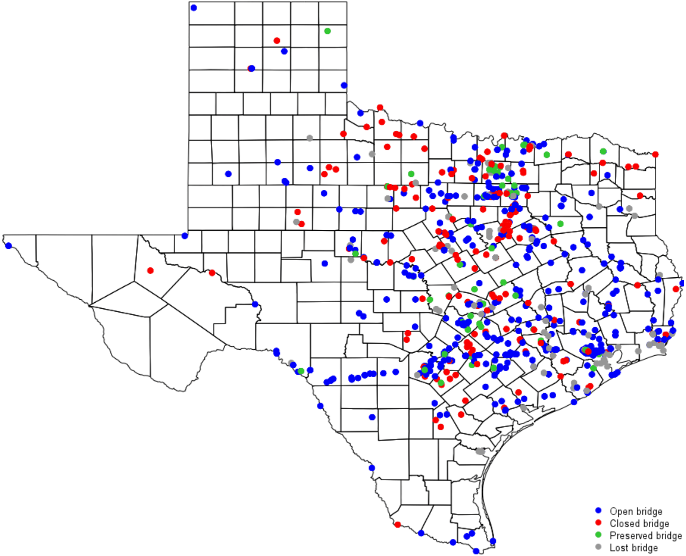
INDICATOR: BCWT - Benefit/Cost Weighted Average (Overall)

YEAR OF FORECAST	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017
Rural Bridges															
Interstate (01)	11.495	10.827	5.694	4.636	3.353	3.010	2.632	2.518	2.333	2.308	2.318	2.326	2.362	2.397	2.428
On NHS	11.495	10.828	5.694	4.636	3.353	3.010	2.633	2.524	2.333	2.307	2.318	2.326	2.372	2.403	2.430
Off NHS	3.814	3.814	3.814	6.030	3.680	3.149	2.492	1.984	2.287	2.430	2.436	2.442	1.960	2.034	2.200
Other Princ. Arterial (02)	18.445	5.899	4.432	3.592	3.128	2.795	2.445	2.202	2.146	2.016	1.991	1.972	1.977	1.962	1.918
On NHS	15.888	5.736	4.450	3.610	3.131	2.790	2.442	2.217	2.138	2.010	1.982	1.969	1.963	1.944	1.930
Off NHS	38.705	6.678	3.786	3.426	3.046	2.897	2.512	2.174	2.690	2.091	2.136	1.990	2.174	2.244	1.871
Minor Arterial (06)	9.704	4.008	3.186	2.628	2.163	1.946	1.785	1.706	1.651	1.581	1.539	1.532	1.530	1.536	1.542
On NHS	6.530	3.512	2.913	2.657	2.525	1.934	1.990	1.752	1.677	1.561	1.610	1.506	1.642	1.645	1.497
Off NHS	9.787	4.026	3.186	2.625	2.163	1.946	1.784	1.706	1.651	1.582	1.538	1.532	1.529	1.536	1.543
Major Collectors (07)	4.899	2.207	1.765	1.590	1.504	1.444	1.416	1.406	1.401	1.402	1.408	1.416	1.430	1.446	1.460
On NHS	2.862	2.614	1.995	1.633	1.544	1.436	1.553	1.545	1.333	1.306	1.563	1.566	1.567	1.566	1.566
Off NHS	4.900	2.207	1.765	1.590	1.503	1.444	1.416	1.405	1.402	1.404	1.407	1.415	1.430	1.446	1.459
Minor Collectors (08)	8.064	2.756	2.105	1.809	1.699	1.673	1.642	1.622	1.627	1.641	1.658	1.678	1.695	1.711	1.726
On NHS	---	2.550	2.312	2.156	1.737	1.739	1.690	1.571	1.639	1.622	1.533	1.608	1.605	1.585	1.599
Off NHS	8.064	2.756	2.105	1.809	1.699	1.673	1.642	1.622	1.627	1.641	1.658	1.678	1.695	1.711	1.727
Locals (09)	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---
On NHS	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---
Off NHS	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---
Rural - Total	9.220	3.875	2.882	2.424	2.094	1.926	1.782	1.704	1.667	1.662	1.609	1.609	1.618	1.626	1.628
On NHS	14.517	6.892	4.739	3.703	3.175	2.818	2.483	2.303	2.152	2.041	2.053	2.042	2.051	2.043	2.042
Off NHS	7.734	3.065	2.339	2.008	1.776	1.663	1.581	1.582	1.514	1.495	1.480	1.491	1.494	1.506	1.522

INDICATOR: BCWT - Benefit/Cost Weighted Average

YEAR OF FORECAST	2003	2004	2005	2006
Rural Bridges				
Interstate (01)	11.495	10.827	5.694	4.636
On NHS	11.495	10.828	5.694	4.636
Off NHS	3.814	3.814	3.814	6.030
Other Princ. Arterial (02)	18.445	5.899	4.432	3.592
On NHS	15.888	5.736	4.450	3.610
Off NHS	38.705	6.678	3.786	3.426
Minor Arterial (06)	9.704	4.008	3.186	2.628
On NHS	6.530	3.512	2.913	2.657
Off NHS	9.787	4.026	3.186	2.625

TxDOT used these analyses both to identify ways of improving allocation of bridge funding, and to perform “what if” scenarios for different budget levels.



MATRIX ALGEBRA, BRIEFLY

A **matrix** is a table of numbers.

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

An $r \times c$ matrix has r rows and c columns; matrices are often written with bold capital letters: $\mathbf{A}$.

A matrix is *square* if it has the same number of rows and columns.

A **vector** is a matrix with only one row or one column, often written with bold lowercase: $\mathbf{x}$.

A row vector: $\begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$

A column vector: $\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$

Matrices of the same dimension can be added or subtracted.

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 6 & 8 \\ 10 & 12 \end{bmatrix}$$

The corresponding elements in the two matrices are added or subtracted.

Multiplying a number by a matrix is called *scalar multiplication*.

$$2\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix} = \begin{bmatrix}2 & 4\\ 6 & 8\end{bmatrix}$$

Multiply each element in the matrix by that number.

Multiplying two matrices is *only* possible if their dimensions are compatible.

The number of *columns* in the first matrix must equal the number of *rows* in the second.

The product matrix has the number of *rows* from the first matrix, and the number of *columns* from the second.

All of the formulas involving matrix multiplication are only valid if the dimensions are compatible. This is assumed from here on.

Examples...

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$ is defined.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \begin{bmatrix} 5 & 6 \\ 7 & 8 \\ 9 & 10 \end{bmatrix}$$ is defined.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$ is NOT defined.

In Excel, the `MMULT` array function can be used to multiply two matrices.

Matrix multiplication is associative: $\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}$

Matrix multiplication is distributive: $\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC}$.

Matrix multiplication is **not** commutative: $\mathbf{AB} \neq \mathbf{BA}$ in general.

The identity matrix $\mathbf{I}$ is all zero, except for ones on the diagonal:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Multiplying by the identity does not change any matrix: $\mathbf{IA} = \mathbf{AI} = \mathbf{A}$.

The inverse of a square matrix $\mathbf{A}$ is the unique matrix $\mathbf{A}^{-1}$ such that $\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$.

The inverse matrix has the same dimensions as the original matrix, and can be calculated with the Excel array function `MINVERSE`.

Not all matrices have inverses, but the ones we will see in this course do. If a linear programming matrix does not have an inverse, there is a redundant constraint which can be removed.

Matrix notation can be used to solve systems of linear equations.

$$\begin{aligned} 4x_1 + 2x_2 &= 3 \\ 6x_1 + x_2 &= 4 \end{aligned}$$

can be written as $\mathbf{Ax} = \mathbf{b}$ where

$$\mathbf{A} = \begin{bmatrix} 4 & 2 \\ 6 & 1 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

To solve this equation, multiply both sides of the equation by $\mathbf{A}^{-1}$:

$$\mathbf{Ax} = \mathbf{b} \iff \mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

This is where the Excel formula `MMULT(MINVERSE(·), ·)` comes from.

SIMPLEX METHOD

This is the standard form of a linear program:

$$
\begin{aligned}
\min_{x_1,\dots,x_n} \quad & \sum_{i=1}^{n} c_i x_i \\
\text{s.t.} \quad & \sum_{i=1}^{n} a_{i1} x_i = b_1 \\
& \vdots \\
& \sum_{i=1}^{n} a_{im} x_i = b_m \\
& x_1, \dots, x_n \geq 0
\end{aligned}
$$

It can be written much more compactly using matrix notation:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{c}\mathbf{x} \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

Here $\mathbf{A}$ is an $m \times n$ matrix, $\mathbf{b}$ and $\mathbf{x}$ are column vectors, and $\mathbf{c}$ is a row vector.

Let $\mathbf{B}$ be the matrix corresponding to the coefficients of basic variables, and $\mathbf{N}$ the matrix corresponding to the coefficients of nonbasic variables.

Then

$$\mathbf{Ax} = \mathbf{b} \iff \mathbf{Bx^B} + \mathbf{Nx^{NB}} = \mathbf{b} \iff \mathbf{Bx^B} = \mathbf{b}$$

because $\mathbf{x^{NB}} = \mathbf{0}$.

So, the values of the basic variables have to be $\mathbf{x^B} = \mathbf{B}^{-1}\mathbf{b}$.

We saw this last week when we were solving a system of linear equations to find the values of the basic variables.

Assume we have chosen basic variables x^B and nonbasic variables x^{NB}, and that these are feasible. Recall that this corresponds to an extreme point of the feasible region.

The simplex method moves from such a point to an adjacent extreme point; this is equivalent to swapping one variable out of the basis, and one variable into the basis.

The key question: if I choose some new variable to become basic, what happens to the value of the objective function?

Assume that x_k is currently nonbasic (so equal to zero). What if we were to set x^k equal to λ instead of zero?

To maintain feasibility, the values of the (current) basic variables will also have to change (from $\mathbf{x^B}$ to $\mathbf{x^B} + \mathbf{d^B}$). Let $\mathbf{A^k}$ be the column of the constraint matrix corresponding to x_k.

We must have $\mathbf{B}(\mathbf{x^B} + \mathbf{d^B}) + \mathbf{A^k}\lambda = \mathbf{b}$.

Solving for $\mathbf{d^B}$ gives $\mathbf{d^B} = -\lambda \mathbf{B^{-1}A^k}$.

As a shorthand, we will write $\Delta \mathbf{x^B} = \mathbf{B^{-1}A^k}$.

So, if we set $x^k \leftarrow \lambda$, what happens to the objective function?

First, the objective must increase by $c^k\lambda$.

The objective will *also* change because the other basic variables are changing; this change is $\mathbf{c^B}(\lambda\Delta\mathbf{x^B}) = \lambda\mathbf{c^B}\mathbf{B^{-1}}\mathbf{A^k}$.

So, the total change in the objective if x^k were to become nonbasic (and equal to λ) is:

$$\lambda(c^k - \mathbf{c^B}\mathbf{B^{-1}}\mathbf{A^k})$$

The term in parentheses is the *reduced cost* of the k-th decision variable.

If a nonbasic variable has a *negative* reduced cost, then adding it to the basis will reduce the value of the objective function. If the reduced cost is positive, adding it to the basis will increase the objective; if zero there will be no change in the objective.

The simplex method chooses any nonbasic variable with a negative reduced cost to enter the basis.

Of course, if x^k enters the basis, something must leave.

Since the objective is linear, if it is beneficial to increase x^k a little bit, it will be beneficial to raise it as high as possible.

The only constraint which could be violated as x^k is increased further and further is a nonnegativity constraint for one of the basic variables.

Any basic decision variable x_i for which $\Delta x_i \geq 0$ will always remain feasible no matter how high x^k is.

Any basic decision variable x_i for which $\Delta x_i < 0$ will become smaller and smaller as x^k is increased; eventually it will hit zero and x^k cannot be increased further.

Thus, $x_i/|\Delta x_i|$ is an upper bound on x^k for any x_i with $\Delta x_i < 0$.

So, $x^k \leftarrow \lambda = \min_{i:\Delta x_i < 0} x_i/|\Delta x_i|$

Simplex method

1. Identify an initial feasible basis $\{x_1^B, \ldots, x_m^B\}$.
2. Calculate the reduced cost $\overline{c^k} = c^k - \mathbf{c^B B^{-1} A^k}$ for each nonbasic decision variable.
3. If all of the reduced costs are nonnegative, the current basis is optimal. **STOP.**
4. Otherwise, choose some nonbasic decision variable x^k with a negative reduced cost to enter the basis.
5. Set $x^k = \min_{i:\Delta x_i < 0} x_i / |\Delta x_i|$; the basic variable leaving the basis is one where this minimum is obtained.
6. Return to step 2.

Example