# xsample: an R function for sampling over- and underdetermined linear inverse problems

Karel Van den Meersche

July 17, 2008

## 1 Introduction

xsample is an R function that instead of optimizing a linear problem, returns a sample set that approaches the probability distribution of the solution.

A general linear problem can be written in matrix notation as: [1]

$$\begin{cases} \mathbf{Ax} = \mathbf{b} \\ \mathbf{Ex} \simeq \mathbf{f} \\ \mathbf{Gx} \geq \mathbf{h} \end{cases} \tag{1}$$

A typical linear model consists of a set of equality constraints that have to be met either exactly or approximately, and a number of inequality constraints. When there are as many independent and consistent equations as there are unknowns, there is one set of parameters that meets all the equations, and the model can be called evendetermined. In other cases, there are more equations than unknowns; there is no exact solution and the model is overdetermined. An approximate solution can then be calculated by minimizing some model cost, such as the sum of squared residuals. Linear regressions are examples of such models.

When there are too few equality constraints, the model is underdetermined, and there exist an infinite number of parameter sets that meet the constraints. In many research areas such as engineering, economics, one is interested in only one optimal solution. Thus, one looks for the simplest solution, or the solution that minimizes cost, or maximizes security, profit, efficiency or other variables. This kind of problems are solved with linear programming and quadratic programming techniques.

Instead of searching one optimal solution, xsample() samples the whole solution space, using a Markov Chain Monte Carlo (MCMC) algorithm.

[1] notations: vectors and matrices are in **bold**; scalars in normal font. Vectors are indicated with a small letter; matrices with capital letter. Indices between brackets indicate elements of vectors (as in $a_{(i)}$) or matrices (as in $A_{(i,j)}$). Rows or columns of matrices are indicated as $\mathbf{A}_{(i,)}$ (rows) or $\mathbf{A}_{(,j)}$ (columns). Indices without brackets ($\mathbf{q_1}$, $\mathbf{q_2}$) indicate vectors that are subsequent in a random walk.

# 2 Method

## 2.1 Step 1: eliminate equality constraints

The elements $x_{(i)}$ of the solution vector $\mathbf{x}$ are not linearly independent; they are coupled through the equations in $\mathbf{Ax} = \mathbf{b}$. They are first linearly transformed to a vector $\mathbf{q}$ for which all elements $q_{(i)}$ are linearly independent. If a vector $\mathbf{p}$ is a particular solution of equation (1), then all solutions $\mathbf{x}$ can be written as:

$$\mathbf{x} = \mathbf{p} + \mathbf{Zq} \tag{2}$$

$\mathbf{Z}$ is an orthonormal matrix obtained e.g. from the QR-decomposition of $\mathbf{A}$ and a basis for the null space of $\mathbf{A}$: $\mathbf{AZ} = \mathbf{0}$ and $\mathbf{Z}^T\mathbf{Z} = \mathbf{I}$.
A particular solution $\mathbf{p}$ can be provided if found easily. If not, a particular solution is calculated using LSEI [1].

There are no equality constraints for the elements in $\mathbf{q}$. The inequality constraints can be rewritten as:

$$\mathbf{G}(\mathbf{p} + \mathbf{Zq}) \geq \mathbf{h} \tag{3a}$$

$$\mathbf{GZq} + (\mathbf{Gp} - \mathbf{h}) \geq \mathbf{0} \tag{3b}$$

The problem of finding $\mathbf{x}$ with equality and inequality constraints, has been translated into the problem of sampling $\mathbf{q}$ with only inequality constraints.
Because $\mathbf{p}$ meets the inequality constraints $\mathbf{Gp} \geq \mathbf{h}$, there is already one trivial solution of $\mathbf{q}$: the null vector $\mathbf{0}$. From this point, it is possible to sample new points.

The probability of $\mathbf{q}$ is a product of the probability of $\mathbf{x}$ and the Jacobian determinant:

$$p(\mathbf{q}) = p(\mathbf{x})||\frac{\partial \mathbf{x}}{\partial \mathbf{q}}|| \tag{4}$$

In this case, the Jacobian is $|\mathbf{Z}| = 1$. Therefore $p(\mathbf{x}) = p(\mathbf{q})$.

## 2.2 Step 2: random walk

### 2.2.1 Markov Chain

The probability distribution of $\mathbf{q}$ can be sampled numerically using a random walk. A Metropolis algorithm [2] produces a series of samples of the solution space of which the distribution approximates the true probability distribution. New samples $\mathbf{q_2}$ are drawn randomly from a jump distribution $j(.|\mathbf{q_1})$ that only depends on the previously accepted point $\mathbf{q_1}$. The new sample point is either accepted or rejected based on the following criterion:

$$\text{if} \quad r \leq \frac{p(\mathbf{q_2})}{p(\mathbf{q_1})} \quad \text{accept } \mathbf{q_2} \text{ else accept } \mathbf{q_1} \tag{5}$$

$r$ is sampled randomly between 0 and 1. The only prerequisite for the sample distribution to converge to the true probability distribution, is that the jump distribution from which a new sample is drawn, is symmetrical: the probability to jump from $\mathbf{q_1}$ to $\mathbf{q_2}$, $j(\mathbf{q_2}|\mathbf{q_1})$, has to be the same as the probability to jump from $\mathbf{q_2}$ to $\mathbf{q_1}$, $j(\mathbf{q_1}|\mathbf{q_2})$.

Three jump algorithms for selecting new samples were implemented: Two hit-and-run algorithms [3] and one algorithm that uses the inequality bounds as reflective planes. . All three algorithms fulfill the symmetry prerequisite for the metropolis algorithm.

### 2.2.2 Random Directions Algorithm (rda) [3]

The algorithm exists of two steps: first it selects a random direction. This direction together with the starting point define a line in solution space. In step 2, it searches the intersections of this line with the planes defined by the inequality constraints. A new point is then sampled uniformly along the line segment that fulfills all inequalities.

### 2.2.3 Coordinates Directions Algorithm (cda) [3]

Very similar to the random directions algorithm, this algorithm starts with selecting a direction along one of the coordinate axes. The rest of the algorithm is analogous to the random directions algorithm. This proves to be very efficient for linear problems.

**NOTE:** rda and cda only work if G and H define a convex solution space. In an open or half open space, these algorithms will spawn error messages.

### 2.2.4 mirror

This algorithm was inspired by the reflections in mirrors and uses the inequality constraints as reflecting planes. New samples are taken from a normal distribution with $\mathbf{q_1}$ as average and a fixed standard deviation, called the jump length. This jump length has a significant influence on the efficiency of the algorithm .

**random walk with inequality constraints** When jumping with a Gaussian jump distribution in a highdimensional space with inequality constraints, the chance to jump out of the accepted range is exponential to the number of inequality constraints. To make sure every new sample fulfills all inequalities, the inequalities are used as mirrors.

In a euclidean space, every inequality constraint defines a boundary of the feasible subspace. Each boundary can be considered a multidimensional plane (a hyperplane). One side of the hyperplane is the feasible range, where the inequality is fulfilled. The other side of the hyperplane is non-feasible. The hyperplanes are determined by the following set of equations:

$$(\mathbf{GZ})_{(,i)}\mathbf{q} + (\mathbf{Gp})_{(i)} - h_{(i)} = 0 \; \forall i \tag{6}$$

If $\mathbf{q_1}$ is a point for which the equality constraints are fulfilled, a new point $\mathbf{q_2}$ can be sampled in the following way: first $\mathbf{q_{2-0}}$ is sampled in the unrestricted space, ignoring all inequality constraints.

$$\mathbf{q_{2-0}} = \mathbf{q_1} + \varepsilon \tag{7}$$

If $\mathbf{q_{2-0}}$ is in the feasible range (all equalities are met), $\mathbf{q_{2-0}}$ is accepted as a sample point and becomes the new starting point $\mathbf{q_1}$ for further sampling.

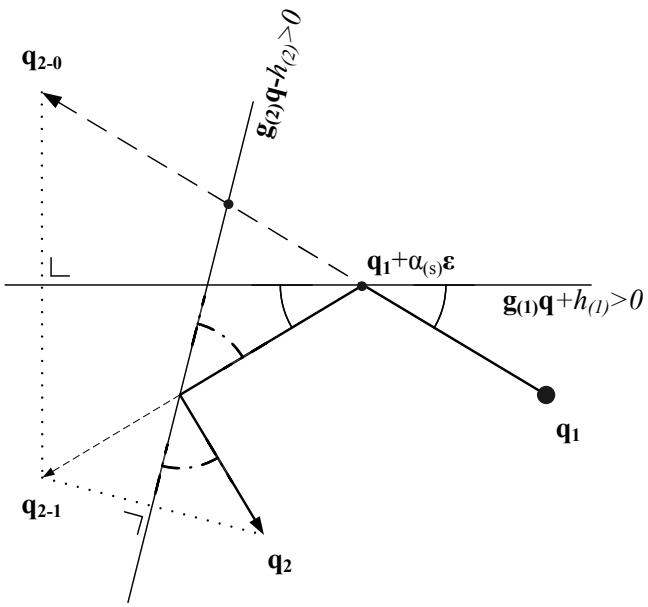


Figure 1: MCMC jump with inequality constraints functioning as mirrors. See text for explanation.

If there are unmet inequalities (figure 1), then the new point $\mathbf{q_{2-0}}$ is mirrored consecutively in the hyperplanes representing the unmet inequalities: the line segment $\mathbf{q_1} \rightarrow \mathbf{q_{2-0}}$ crosses these hyperplanes. For each hyperplane, a scalar $\alpha_{(i)}$ can be calculated for which

$$(\mathbf{GZ})_{(,i)}(\mathbf{q_1} + \alpha_{(i)}\varepsilon) + (\mathbf{Gp})_{(i)} - h_{(i)} = 0 \tag{8}$$

with $\varepsilon = \mathbf{q_{2-0}} - \mathbf{q_1}$. The hyperplane with the smallest non-negative $\alpha_{(i)}$, call it $\alpha_{(s)}$, is the hyperplane that is crossed first by the line segment. $\mathbf{q_{2-0}}$ is mirrored around this hyperplane. If the new point ($\mathbf{q_{2-1}}$ in figure 1) still has unmet inequalities, a new set of $\alpha_{(i)}$'s is calculated from the line segment between the new point and the intersection of the previous line segment and the first hyperplane: $\mathbf{q_1} + \alpha_{(s)}\varepsilon$.
$\mathbf{q_{2-1}}$ is again reflected in the hyperplane with smallest non-negative $\alpha_{(i)}$. This is repeated until all inequalities are met. The resulting point $\mathbf{q_2}$ is in the feasible subspace and is accepted as a new sample point.

# References

[1] Haskell, K.H., Hanson, R.J.: An algorithm for linear least-squares problems with equality and non-negativity constraints. Mathematical Programming **21**(1), 98-118 (1981).

[2] Roberts, G.: Markov chain concepts related to sampling algorithms. In: W. Gilks, S. Richardson, D. Spiegelhalter (eds.) Markov Chain Monte Carlo in practice, pp. 45-58. Chapman & Hall (1996)

[3] Smith, R.L.: Efficient Monte Carlo Procedures for Generating Points Uniformly Distributed over Bounded Regions. Operations Research **32**, pp. 1296-1308 (1984).