IterativeLinearSolvers

IterativeLinearSolvers

It is not uncommon to have to solve (large) linear systems for which it is both expensive and unnecessary to explicitly define all coefficients of the underlying matrix.

In the present approach, the only requirement we make on the linear operator A is the ability to compute the matrix-vector product A.x (and possibly the product A'.x, where A' is the transpose of A).There are quite a few iterative solvers around for addressing such problems: conjugate gradient, SYMMLQ, to name but a few. This proposal aims at including these algorithms into commons-math. I have already implemented some of these algorithms, but I would like to have a discussion on the interfaces (as well as the class/method names). Here are a few ideas.

The LinearMap interface

The linear operator A would be defined as implementing the interface LinearMap, which could look like this

```
public interface LinearMap extends AnyMatrix{
  public void apply(double[] x, double[] y);
  public void apply(RealVector x, RealVector y);
}
```

Comments on AnyMatrix

Since it extends AnyMatrix, a LinearMap must implement getColumnDimension() and getRowDimension(). There might be a conflict in terminology, since the whole point in LinearMap is to forget about the underlying matrix, and to focus on the linear operator. Maybe it would be better to define radically different functions, for example

```
getDomainDimension()
```

```
getCodomainDimension()
```

If this second option is adopted, then maybe the filiation with AnyMatrix should be given up altogether.

Comments on the parameters of apply

In order to avoid memory allocation, the image y = A.x is passed as a parameter. I think this is generally more efficient, but I might be wrong. Any thoughts?

Exceptions thrown by the iterative solvers

Quite a few exceptions might be thrown during the iterations. Typical cases are

- non-symmetric linear map,
- non-positive definite linear map.

Of course, existing exceptions handle those cases for ***matrices***. For linear map, occurence of the corresponding errors is not detected explicitely. For example, it is never checked that Aii > 0 (since coefficients of the LinearMap cannot be accessed). Rather, an exception might be thrown if during the iterations, x'.A.x < 0 (primed vector/matrices = transpose).

NonSymmetricMatrixException, NonPositiveDefiniteMatrixException are not well suited for this (there is no appropriate constructor, and the error messages are ill adapted). I was initially thinking of modifying those exceptions, but I am now leaning towards the definition of new exceptions. However, I think this second solution is not really satisfactory either, since it multiplies the number of different exceptions.

A quite nice option would be to define a NonPositiveDefiniteLinearMapException, with quite general error message, and to have NonPositiveDefiniteMatrixException extend this exception, with more specific error messages, better suited to matrices whose coefficients can be accessed.

Again, feedback would be appreciated!