The Visual Computer manuscript No.
(will be inserted by the editor)

On the modified conjugate gradient method in cloth simulation

Uri M. Ascher[1], **Eddy Boxerman**[2] ⋆

Department of Computer Science,
University of British Columbia,
Vancouver, BC, V6T 1Z4, Canada.
e-mail: {ascher|eddybox}@cs.ubc.ca
www.cs.ubc.ca/∼{ascher|eddybox}

Received: date / Revised version: date

⋆ Supported in part under NSERC Research Grant 84306.
Correspondence to: Uri Ascher
ascher@cs.ubc.ca
Phone: (604) 822-4907; Fax: (604) 822-5485

Abstract The seminal paper on cloth simulation by Baraff & Witkin (1998) presents a modified preconditioned conjugate gradient (MPCG) algorithm for solving certain large, sparse systems of linear equations. These arise when employing implicit time integration methods aimed to achieve large step cloth simulation in the presence of constraints.

The present paper improves the robustness and efficiency of this MPCG algorithm. We prove convergence. For this, we recast the algorithm into a linear algebra setting, identifying its filtering procedure as an orthogonal projection. This leads not only to a convergence proof but also to a correction in the initiation stage of the original algorithm which improves its efficiency. We give an example to illustrate the performance improvement offered by this correction.

Key words Cloth simulation – constraints – orthogonal projection – conjugate gradients – implicit time stepping

1 Introduction

The paper of Baraff & Witkin [4] proposes the use of a semi-implicit time discretization method for cloth simulation and develops various algorithmic details. Specifically, a backward Euler discretization is utilized (see, e.g., [2]), approximated by the use of one approximate Newton iteration for the resulting nonlinear equations at each time step.[1] This allows taking much larger step sizes than possible using explicit time discretizations, which is useful when there are no frequent collisions [9]. The reason that explicit schemes are then limited is numerical stability, which is due to the stiffness caused by the high resistance of cloth to stretching (unlike the resistance to shearing and bending). The paper [4] has justifiably been very influential (see, e.g., [7,8,6] and references therein) and boasts beautiful practical results.

The semi-implicit integration method necessitates solving a linear system of equations at each time step. This system can be large and must be solved rapidly for interactive performance. In the absence of constraints, Baraff & Witkin [4] therefore design a model which yields a symmetric positive definite system, and they apply a simply preconditioned conjugate gradient (PCG) method for its approximate solution.

Constraints have to be accommodated, however. This modifies the system of equations to be solved. They therefore incorporate constraints of certain types directly into the equations and devise a corresponding *modified* preconditioned conjugate gradient method (MPCG). This algorithm has been subsequently used by others, see e.g. the recent paper by Choi & Ko [7], even though no proof, or guarantee, has been given hitherto for its convergence.

Our contributions are these:

- Proof of convergence. This is done as follows. In order to put the MPCG on a sounder theoretical footing we recast the algorithm in a linear algebra setting. The crucial observation is that the filtering procedure proposed in [4] is an *orthogonal projection* (see, e.g., [10]). This then allows us to relate the MPCG to the usual preconditioned conjugate gradient method applied to a restricted, unconstrained, positive definite system. The established connection then provides a proof of convergence for the MPCG algorithm, as shown in Section 4.
- An improvement over the initialization procedure proposed in [4] results from our approach. This is expressed in the choice of initial iterate, as well as in the selection of a convergence criterion. In Section 5 we give a cloth simulation example that demonstrates the potential performance improvements gained.

2 The linear algebra problem

Throughout this paper we use boldfaced lowercase roman letters to denote vectors and uppercase roman letters for matrices.

Consider the cloth as a bunch of particles in $3D$, i.e. each particle i has 3 coordinates $\mathbf{q}_i,\ i = 1, \ldots, N$. The equations of motion are assembled, accounting for various forces such as stretching, shearing, bending, gravity, air-drag and damping. The time-dependent differential system is large and stiff. When large time steps are possible (i.e. in the absence of other reasons to make them small) a discretization method with stiff decay [2] is advantageous, at least for the stretching energy term. Baraff & Witkin [4] use the backward Euler method, but only one approximate Newton iteration is applied towards the solution of the resulting nonlinear system of equations. This works when the step size is not too large (it is well-known that Newton's method is only guaranteed to converge locally, so stability is not entirely unconditional), yet the resulting practical step size restriction is typically much larger than what is possible with explicit time stepping. The resulting semi-implicit time stepping scheme yields a linear system of size $n = 3N$ for the *unconstrained problem*,

$$A\mathbf{x} = \mathbf{b}. \tag{1}$$

The authors take extra pains to ensure that the matrix A be symmetric positive definite by dropping some Jacobian terms (which is one reason why the Newton iteration is only approximate).

The unknowns $\mathbf{x} \in \Re^{3N}$ in [4] are velocity changes of the particles over a time step. But a similar system may be obtained where the unknowns are position changes (e.g., Choi & Ko [7]) and everything generalizes to the latter choice. The cloth simulation example in Section 5 uses the modelling technique of the latter paper.

Next, there are constraints (cloth-solid), which are expressed as direct values of no change, or of a specified change, in the velocities (resp. positions) in certain directions for each particle. Thus, for each particle (node) i let $ndof(i)$ denote

[1] This is also occasionally referred to as a *linearly implicit* method.

its number of degrees of freedom: if $ndof(i) = 3$ then there are no constraints on this particle; if $ndof(i) = 2$ then there is one direction $\boldsymbol{\xi}_i$ ($|\boldsymbol{\xi}_i| = 1$) of prohibited motion; if $ndof(i) = 1$ then there are two mutually orthogonal directions $\boldsymbol{\xi}_i$ and $\boldsymbol{\eta}_i$ ($|\boldsymbol{\xi}_i| = 1,\ |\boldsymbol{\eta}_i| = 1,\ \boldsymbol{\xi}_i^T\boldsymbol{\eta}_i = 0$) of prohibited motion; if $ndof(i) = 0$ then all motion is prohibited for this particle. Define,

$$S_i = \begin{cases} I & ndof(i) = 3 \\ I - \boldsymbol{\xi}_i\boldsymbol{\xi}_i^T & ndof(i) = 2 \\ I - \boldsymbol{\xi}_i\boldsymbol{\xi}_i^T - \boldsymbol{\eta}_i\boldsymbol{\eta}_i^T & ndof(i) = 1 \\ 0 & ndof(i) = 0 \end{cases}$$
$$S = diag\{S_1, \ldots, S_N\}. \tag{2}$$

Note that S_i is an *orthogonal projection*, hence so is the filter S. This is a crucial observation. Recall that an orthogonal projection matrix S satisfies

$$S^T = S^2 = S, \quad 0 \le rank(S) = r \le n.$$

It can be used to decompose vectors as a direct sum. Thus, if

$$\mathbf{x} = \mathbf{u} + \mathbf{v}, \quad \mathbf{u} = S\mathbf{x}, \mathbf{v} = (I - S)\mathbf{x},$$

then $\mathbf{u}$ and $\mathbf{v}$ are orthogonal, $\mathbf{u}^T\mathbf{v} = 0$.

Here, the *constrained problem* can be written in terms of the projection matrix S defined in (2) and a given vector $\mathbf{z}$ of dimension n (like $\mathbf{x}'s$) such that the problem is

$$SA\mathbf{x} = S\mathbf{b}, \tag{3a}$$
$$(I - S)\mathbf{x} = (I - S)\mathbf{z}. \tag{3b}$$

In words, for each particle the equations of motion hold only in the subspace projected by S, $range(S)$, whereas in the subspace $range(I - S)$ the given values of $\mathbf{z}$ determine velocity or position changes. As the two subspaces are orthogonal the two types of motion are separated. This is all written in the form (3).

3 Modified conjugate gradient method

The method proposed in Baraff & Witkin [4] for the numerical solution of (3) consists of starting from $\mathbf{x}_0 = \mathbf{z}$ and then applying modified PCG iterations, where the modification from usual consists of restricting the iteration to $range(S)$. Thus, any iterate $\mathbf{x}_k$ can be written as

$$\begin{aligned}\mathbf{x}_k &= S\mathbf{x}_k + (I - S)\mathbf{x}_k = S\mathbf{x}_k + (I - S)\mathbf{x}_0 \\ &= S\mathbf{x}_k + (I - S)\mathbf{z}. \end{aligned} \tag{4}$$

Clearly, then, $(I - S)\mathbf{x}_k = (I - S)\mathbf{z}$.

Here is our version of the MPCG algorithm. Our choices of the initial iterate $\mathbf{x}$ and of b_δ are different from those in [4]. These improvements are discussed in Section 5. For a given tolerance tol on the relative residual, do:

1. Set the initial iterate as follows: Let $\mathbf{y}$ be a natural guess for the case of no constraints. (In the present context, one chooses $\mathbf{y} = \mathbf{0}$ or the result from the previous time step.) Then set the initial iterate to be
$$\mathbf{x} = \mathbf{x}_0 = S\mathbf{y} + (I - S)\mathbf{z}.$$
2. Initialize
$$\begin{aligned}\hat{\mathbf{b}} &= S(\mathbf{b} - A(I - S)\mathbf{z}), \\ b_\delta &= \hat{\mathbf{b}}^T P^{-1}\hat{\mathbf{b}}, \\ \mathbf{r} &= S(\mathbf{b} - A\mathbf{x}) \\ \mathbf{p} &= SP^{-1}\mathbf{r} \\ \delta &= \mathbf{r}^T\mathbf{p}. \end{aligned}$$
Here, S is the filter of (2), P is the preconditioner, $\mathbf{r}$ is the residual, $\mathbf{p}$ is the PCG direction, and δ is a squared "energy norm" of the residual comparable to the corresponding squared "energy norm" of the constrained right hand side, b_δ.
3. While $\delta > tol^2\ b_\delta$,
$$\begin{aligned}\mathbf{s} &= SA\mathbf{p} \\ \alpha &= \delta/(\mathbf{p}^T\mathbf{s}) \\ \mathbf{x} &= \mathbf{x} + \alpha\mathbf{p} \\ \mathbf{r} &= \mathbf{r} - \alpha\mathbf{s} \\ \mathbf{h} &= P^{-1}\mathbf{r} \\ \hat{\delta} &= \delta \\ \delta &= \mathbf{r}^T\mathbf{h} \\ \mathbf{p} &= S(\mathbf{h} + (\delta/\hat{\delta})\mathbf{p}) \end{aligned}$$

It can be readily verified that upon setting $S = I$ the usual PCG algorithm is obtained from MPCG.

4 Proof of convergence

Consistently good performance is reported in [4] and [7] using the MPCG (without our corrections). However, a proof of convergence and a specific set of experiments do not amount to quite the same thing. Below we prove that the MPCG indeed converges.

The fact that S is an orthogonal projection allows us to write the direct sum in (4) in a unique way. It also allows to postulate the existence of a matrix $U \in \Re^{n\times r}$ having r orthonormal columns, such that

$$S = UU^T.$$

Note that $U^TU = I$ and $rank(S) = r$. We never actually form the matrix U in practice, but we use its existence.

Therefore, we can write

$$\mathbf{x} = S\mathbf{x} + (I - S)\mathbf{x} = UU^T\mathbf{x} + (I - S)\mathbf{z}.$$

Substituting this into (3a) and multiplying by U^T yields the unconstrained, positive definite system

$$\tilde{A}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}, \tag{5}$$

where

$$\tilde{A} = U^T A U,$$
$$\tilde{\mathbf{b}} = U^T(\mathbf{b} - A(I-S)\mathbf{z}),$$
$$\tilde{\mathbf{x}} = U^T\mathbf{x}.$$

The linear system of size r in (5) has a symmetric positive definite matrix. The conjugate gradient method for (5) therefore provably converges; see, e.g., [10,3]. Our contention is that our corrected version of the MPCG algorithm of [4] is equivalent to the usual PCG method applied to (5). Upon showing this we will have proved convergence for the MPCG algorithm.

For this purpose, define

$$\tilde{\mathbf{p}} = U^T\mathbf{p},$$
$$\tilde{\mathbf{r}} = U^T\mathbf{r},$$
$$\tilde{\mathbf{s}} = U^T\mathbf{s},$$
$$\tilde{P}^{-1} = U^T P^{-1} U.$$

Note that for any n-vector $\mathbf{w} \in range(S)$, $\mathbf{w} = S\mathbf{w}$. Hence, defining the r-vector $\tilde{\mathbf{w}} = U^T\mathbf{w}$ also implies the inverse transformation

$$\mathbf{w} = S\mathbf{w} = U\tilde{\mathbf{w}}.$$

By definition, $\mathbf{p}, \mathbf{r}, \mathbf{s} \in Range(S)$, so they all have such inverse transformations.

Initially

1. The setting of the initial iterate in MPCG implies the initial iterate for $\tilde{\mathbf{x}}$,
$$U^T\mathbf{x} = \tilde{\mathbf{x}}_0 = U^T\mathbf{x}_0.$$
2. The definition of b_δ implies
$$b_\delta = \tilde{\mathbf{b}}^T\tilde{P}^{-1}\tilde{\mathbf{b}}.$$
3. $\tilde{\mathbf{r}} = U^T\mathbf{r} = U^T(\mathbf{b} - A\mathbf{x}) = U^T(\mathbf{b} - A(I-S)\mathbf{z} - AU\tilde{\mathbf{x}}) = \tilde{\mathbf{b}} - \tilde{A}\tilde{\mathbf{x}}$.
4. $\tilde{\mathbf{p}} = U^T\mathbf{p} = \tilde{P}^{-1}\tilde{\mathbf{r}}$.
5. $\delta = \mathbf{r}^T\mathbf{p} = \tilde{\mathbf{r}}^T\tilde{P}^{-1}\tilde{\mathbf{r}}$.

The resulting values for δ and b_δ provide for a sensible convergence criterion when comparing δ to $tol^2\ b_\delta$.

At each iteration

1. $\tilde{\mathbf{s}} = U^T A\mathbf{p} = \tilde{A}\tilde{\mathbf{p}}$.
2. $\alpha = \delta/(\mathbf{p}^T\mathbf{s}) = \delta/(\tilde{\mathbf{p}}^T U^T A\mathbf{p}) = \delta/(\tilde{\mathbf{p}}^T\tilde{A}\tilde{\mathbf{p}}) = \delta/(\tilde{\mathbf{p}}^T\tilde{\mathbf{s}})$.
Hence, both α and δ are the same in the MPCG algorithm as they would be in the usual PCG algorithm for (5).
3. $\tilde{\mathbf{x}} = \tilde{\mathbf{x}} + \alpha\tilde{\mathbf{p}}$.
Here, as in all previous relevant lines, there is a unique, well-defined inverse transformation from $\tilde{\mathbf{x}}$ to $\mathbf{x}$.
4. $\tilde{\mathbf{r}} = \tilde{\mathbf{r}} - \alpha\tilde{\mathbf{s}}$.
5. $\tilde{\mathbf{h}} = U^T\mathbf{h} = U^T P^{-1}\mathbf{r} = U^T P^{-1} U\tilde{\mathbf{r}} = \tilde{P}^{-1}\tilde{\mathbf{r}}$.
6. $\delta = \mathbf{r}^T\mathbf{h} = \tilde{\mathbf{r}}^T\tilde{P}^{-1}\tilde{\mathbf{r}} = \tilde{\mathbf{r}}^T\tilde{\mathbf{h}}$.
7. $\tilde{\mathbf{p}} = U^T(\mathbf{h} + (\delta/\hat{\delta})\mathbf{p}) = \tilde{\mathbf{h}} + (\delta/\hat{\delta})\tilde{\mathbf{p}}$.

The usual PCG algorithm for (5) has thus been shown to be equivalent to the slightly corrected MPCG algorithm of Baraff & Witkin [4]. Hence the latter converges with the usual convergence performance of the PCG algorithm for the symmetric positive definite system (5).

5 Examples

In the original MPCG algorithm [4] the convergence criterion uses $b_\delta = (S\mathbf{b})^T P(S\mathbf{b})$. We have modified this to $b_\delta = (S(\mathbf{b} - A(I-S)\mathbf{z}))^T P^{-1}(S(\mathbf{b} - A(I-S)\mathbf{z}))$ because the latter is more directly comparable to δ, as is evident from the proof in Section 4. For the experiment described below we tried both values for b_δ. While this difference can be absorbed by modifying the value of the error tolerance, our definition of b_δ is the natural one.

More importantly, the original MPCG algorithm uses the initial iterate $\mathbf{x} = \mathbf{z}$, whereas we advocate $\mathbf{x} = S\mathbf{y} + (I-S)\mathbf{z}$. This allows us to use the solution from the previous time step as our inital guess for $\mathbf{x}$, while still maintaining the constraints $\mathbf{z}$. Next, we construct such an experiment with the sole purpose of highlighting this difference and studying its effect on the effective convergence rate of the MPCG algorithm.

Our experiment involves one square sheet of cloth and no collisions, where each of the four corners are constrained to move sinusoidally in time in the vertical direction. See Figure 1.

The modeling approach of Choi & Ko [7] is used, where $\mathbf{x}$ in (3) stands for changes in particle positions. The vector $\mathbf{z}$ therefore involves the derivative of this sinusoidal forcing function, given by $a\sin(\omega t)$ with various values chosen for amplitude a and frequency ω.

We use the semi-implicit backward Euler method with the constant time step $\Delta t = 0.05$, and set $tol = 0.01$. A block diagonal preconditioner $P = diag\{A_{ii}\}_{i=1}^N$ is applied, where A_{ii} is the ith 3×3 block on the diagonal of A.

Using this technique, we have observed an improvement in the average number of MPCG iterations (over 100 time steps), generally requiring only 45% to 75% of that required by the original algorithm. We have found this to hold true upon variation of the following parameters:

- stiffnesses (stretch, shear, bend)
- cloth density
- amplitude and frequency of the forcing function
- constraint configurations (sheet over square and round tables, two corners, etc.)
- mesh size (number of particles, distance between particles)
- preconditioner (diagonal and block-diagonal parts of A)
- discretization method (backward Euler [4] and BDF2 [7])
- time step size

Specific iteration counts for varying densities and error tolerances tol are given in Tables 1 and 2. Varying density has

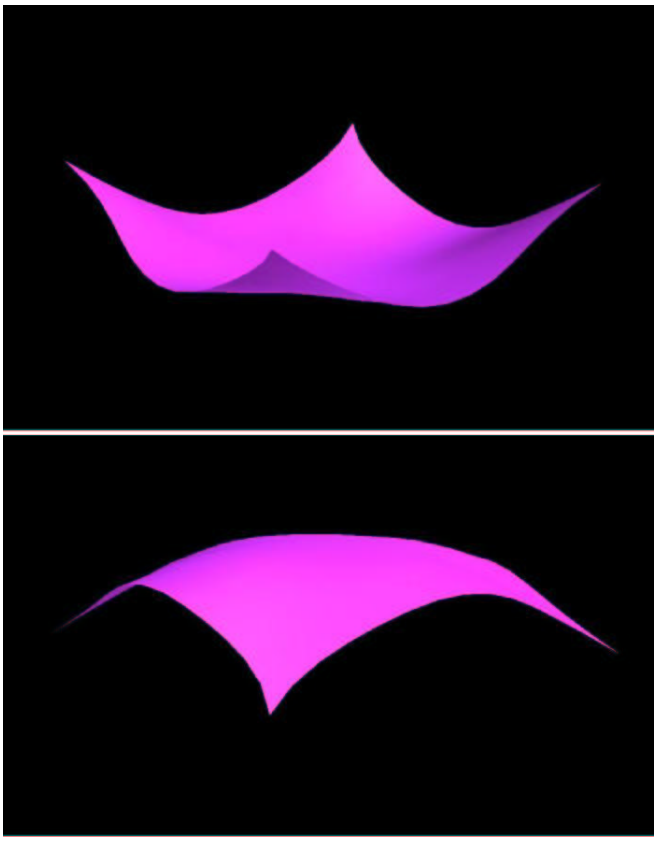

Fig. 1 Snapshots from animation: four pinned ends are constrained to move sinusoidally up and down.

a strong influence on convergence, with high densities causing A to approach the identity matrix. And of course decreasing the error tolerance tol causes an increase in the number of iterations. The point here is that regardless of the iteration count, the improvement *ratio* holds.

Density (kg/m^2)	**# Iterations**	
	Original MPCG	**Corrected MPCG**
0.01	27	19
0.1	22	15
1.0	16	10
10	11	6
100	6	3

Table 1 MPCG Iteration Counts vs. Cloth Density

Tolerance tol	**# Iterations**	
	Original MPCG	**Corrected MPCG**
0.1	22	15
0.01	22	15
0.001	30	22
0.0001	45	31
0.00001	60	43

Table 2 MPCG Iteration Counts vs. Convergence Tolerance

6 Conclusions and further comments

Viewing the MPCG algorithm in a linear algebra setting identifying the filtering procedure of [4] as an orthogonal projection, we have improved both robustness of this algorithm by proving convergence and performance efficiency by using a better initialization.

The effect of our improved initialization depends on the application. Specifically, the more the given constraint $\mathbf{z}$ differs from $\mathbf{y}$ the more pronounced is the improvement. This effect is clearly demonstrated in Section 5. (Of course, if $\mathbf{z} = \mathbf{y}$ then $S\mathbf{y} + (I - S)\mathbf{z} = \mathbf{z}$ and there is no improvement at all.)

Below we make some additional remarks which lead beyond the scope of the present article.

1. The purpose of the preconditioner is to cluster the eigenvalues of $\tilde{P}^{-1}\tilde{A}$ tightly. How do we achieve this with P? Note that
$$\tilde{P}^{-1}\tilde{A} = U^T P^{-1} S A U.$$
So, the spectrum of $\tilde{P}^{-1}\tilde{A}$ is different from the spectrum of $P^{-1}A$. In [4] the authors used $P = diag\{A\}$. In [7] the authors used $P = diag\{A_{ii}\}_{i=1}^{N}$. The latter also experimented with SOR and ILU [11,5], but without much additional success. Our experiment reported in Section 5 also did not display much sensitivity to the choice of preconditioner. Perhaps this is because the systems considered for rapid animation cannot be too detailed, hence the eigenvalue spectra are not as wide as they typically are when elliptic partial differential equations are approximated on a fine grid by a finite element or finite volume method [3,1]. Also, it is possible that treating A with P as if there are no constraints is less effective when constraints modify the system into (3). Applying these standard preconditioners to SA rather than to A could be better, because the eigenvalues of $\tilde{P}^{-1}\tilde{A}$ are the same as the nonzero eigenvalues of $P^{-1}SA$. However, note that then the matrix "P^{-1}" may be singular.
2. In the derivation of the unconstrained equations (1) in both [4] and [7] the authors drop a term in the Jacobian of the damping forces without any physical justification (according to them), just to keep A symmetric. But the problem (3) can be written as a system of linear equations
$$\begin{aligned} B\mathbf{x} &= \mathbf{c}, \quad \text{where} \\ B &= SA + (I - S), \\ \mathbf{c} &= S\mathbf{b} + (I - S)\mathbf{z}. \end{aligned} \tag{6}$$
Some preconditioned Krylov-space method (see, e.g., [11]) can be applied to (6), where B no longer needs to be strictly symmetric positive definite.

What Krylov-space method would be particularly good for (6)? How would this method compare against the MPCG method given above? One idea is to consider applying (preconditioned) BICGstab to the unconstrained problem (1), where now A is no longer symmetric positive definite, but one hopes it is not far from such a matrix; see [11, 5] and Section 5 of [1]. Then modify this method for the constrained case precisely as outlined above for the PCG method. It is unclear a priori how this approach would compare to attacking (6) directly – this may well depend on the specifics of an application of cloth animation and is left for future work.

Acknowledgements We wish to thank Xavier Granier for going over this manuscript and making useful suggestions.

References

1. D. Aruliah, U. Ascher, E. Haber, and D. Oldenburg. A method for the forward modelling of 3D electromagnetic quasi-static problems. *Math Models and Methods in Appl. Science*, 11:1–21, 2001.
2. U. Ascher and L. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, Philadelphia, PA, 1998.
3. O. Axelsson and V.A. Barker. *Finite Element Solution of Boundary Value Problems*. AP, 1984.
4. D. Baraff and A. Witkin. Large steps in cloth simulation. In *SIGGraph*, pages 43–54. ACM, 1998.
5. R. Barrett, M. Berry, T.F. Chan, J. Demmel, , J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, Philadelphia, 1994.
6. R. Bridson, R. Fedkiw, and J. Anderson. Robust treatment of collisions, contact and friction for cloth animation. In *SIGGraph*. ACM, 2002.
7. K-J Choi and H-S Ko. Stable but responsive cloth. In *SIGGraph*. ACM, 2002.
8. F. Cordier and N. Magnenat-Thalmann. Real-time animation of dressed virtual humans. In *Eurographics*. Blackwell Publishers, 2002.
9. M. Courshesnes, P. Volino, and N. Magnenat-Thalmann. Versatile and efficient techniques for simulating cloth and other deformable objects. In *SIGGraph*, pages 137–144. ACM, 1995.
10. G.H. Golub and C.F. van Loan. *Matrix Computations*. Johns Hopkins University Press, 1988.
11. Y. Saad. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company, 1996.