Early Termination of Conjugate Gradients for Corotated Finite Elements

Alex Dahl
adahl1@umbc.edu
University of Maryland, Baltimore County
Baltimore, Maryland

Adam Bargteil
adamb@umbc.edu
University of Maryland, Baltimore County
Baltimore, Maryland

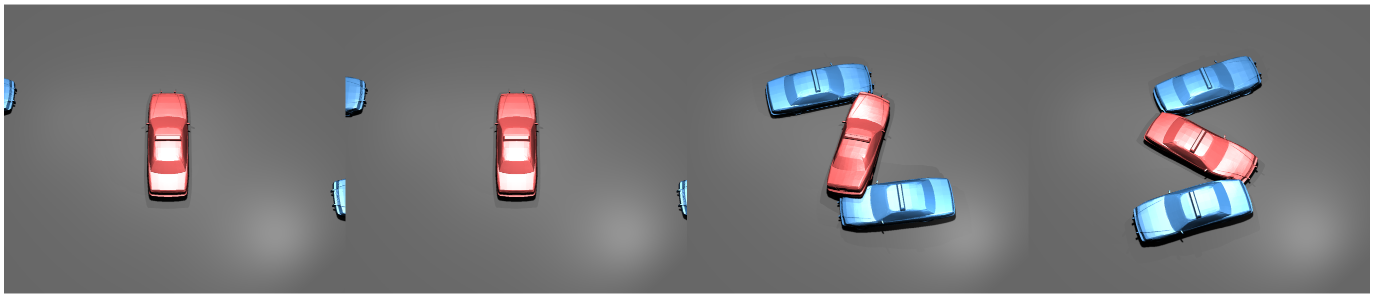

Figure 1: Images from an example scene with conjugate gradient iterations limited to 1, 2, 5, 10. An initial guess of 0 is used.

ABSTRACT

Since the introduction of the conjugate gradient method to computer graphics, researchers have largely treated it as a black box. In particular, an arbitrary small value is chosen for the tolerance and the method is run to convergence. In the context of soft body animation, this approach results in significant wasted computation and has led researchers to consider alternative, more complex, and less versatile approaches. In this paper we argue that in the context of corotational finite elements, less than 10 iterations can give a *good enough* solution and substantial savings of computational cost. We examine the use of different preconditioners for conjugate gradient including the mass and Jacobi matrices, as well as the use of different initial guesses. We show that for our examples an initial guess of the previous velocity and the Jacobi preconditioner works best.

CCS CONCEPTS

• **Computing methodologies** → **Procedural animation**; **Physical simulation**.

KEYWORDS

conjugate gradients, finite element method, computer animation, physics simulation

ACM Reference Format:
Alex Dahl and Adam Bargteil. 2019. Early Termination of Conjugate Gradients for Corotated Finite Elements. In *Motion, Interaction and Games (MIG '19), October 28–30, 2019, Newcastle upon Tyne, United Kingdom.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3359566.3360080

1 INTRODUCTION

Since the introduction of the conjugate gradient method [Shewchuk 1994] to computer graphics by Baraff and Witkin [1998] more than twenty years ago, researchers have largely treated it as a black box. In particular, an arbitrary small value is chosen for the tolerance and the method is run to convergence[1], perhaps with a maximum of 100 or 1000 iterations. This approach achieves good solutions, dissuading researchers from giving further thought to the method's behavior. However, in the context of soft body animation, this naïve approach results in significant wasted computation and has led researchers to consider alternative, more complex, and less versatile approaches. In this paper we ask the question "how many iterations are required to achieve a *good* (enough) solution?" In the context of corotational finite element simulations of soft bodies using linearly implicit Euler time integration we find that, for most examples, only a couple iterations are necessary to achieve good results. For some examples, a single iteration is sufficient. Ten iterations is typically overkill. This finding saves significant computation time and makes the conjugate gradient method competitive with other techniques.

As with all optimization techniques, choosing a good starting point is critical to reaching a good solution quickly. We experimented with three initial guesses for the solution and find that the best initial guess for the velocity at time $t + \Delta t$ is the velocity at time t. We also experimented with several preconditioners and provide some analysis and intuition of the behavior of the conjugate gradient method with different formulations of the linear system. We also show that a similar philosophy can be used to reduce the computation time for the polar decompositions required

[1]Typically, researchers follow Shewchuk [1994] and define convergence to be when the 2-norm of the residual falls below some tolerance, either absolute or relative. In our examples, we use the relative criterion $\mathbf{r} \cdot \mathbf{r} \leq (1\text{e-}10) * \mathbf{b} \cdot \mathbf{b}$, where $\mathbf{r}$ is the residual and $\mathbf{b}$ is the right hand side of the linear system.

for corotational elasticity. The end result is a system for solving elasticity problems with corotational finite elements that is considerably faster than a naïve alternative and is competitive with more complicated state-of-the-art techniques that often come with additional limitations.

2 RELATED WORK

A detailed discussion of conjugate gradients is beyond the scope of this paper, we refer the reader to Shewchuk's [1994] excellent introduction to the topic.

Corotational finite elements were introduced to graphics by Müller and colleagues [Müller et al. 2002; Müller and Gross 2004]. The method requires less computation than non-linear strain finite elements [O'Brien and Hodgins 1999] and avoids the severe distortions of linear elasticity [Hauser et al. 2003]. The method remains popular and is often employed with linearly implicit Euler time integration, which is solved with a conjugate gradient method [Parker and O'Brien 2009]. This approach is simple and versatile, handling a wide variety of scenarios with few limitations and very little precomputation.

Recently, researchers have developed faster techniques that make use of precomputation of matrix decompositions or clever preconditioners [Bouaziz et al. 2014; Brandt et al. 2018; Dinev et al. 2018; Hecht et al. 2012; Liu et al. 2013; Narain et al. 2016; Wang 2015; Wang and Yang 2016]. Some of these approaches place limits on material models or other aspects of the animation. We take a different approach; we adopt the classic approach of linearly implicit Euler time integration of corotational finite elements with the conjugate gradient method. However, instead of running conjugate gradients to convergence, we show that good results are obtained after a couple iterations. We take the same approach to computing the polar decompositions that form the basis of corotational finite elements, finding that the decomposition is typically "symmetric enough" after a single iteration. Our approach does not require any precomputation and has the advantages of simplicity and extensibility—incorporating additional effects such as fracture or plasticity does not require any special treatment beyond updating the element stiffness matrices.

3 METHOD

3.1 Corotational Finite Elements

For completeness, in this subsection we briefly review the corotational strain formulation of finite elements. We largely follow the approach of Parker and O'Brien [2009]. Those familiar with corotational finite elements may safely skip this subsection.

We assume that the deformable body is decomposed into a finite set of elements—disjoint tetrahedra. Tetrahedra share faces, edges, and vertices, the latter of which we refer to as nodes. In *material* space, the nodes have positions, $\mathbf{u}_i$. The nodes also have world space positions, $\mathbf{x}_i$, and velocities, $\mathbf{v}_i$. Let $\mathbf{D}_u$ be the matrix composed of vectors along the edges of a tetrahedron,

$$\mathbf{D}_u = \begin{pmatrix} (\mathbf{u}_1 - \mathbf{u}_0) & (\mathbf{u}_2 - \mathbf{u}_0) & (\mathbf{u}_3 - \mathbf{u}_0) \end{pmatrix}. \tag{1}$$

Let $\mathbf{D}_x$ be defined similarly. Then, the deformation gradient is given by

$$\mathbf{F} = \mathbf{D}_x \mathbf{D}_u^{-1}. \tag{2}$$

For corotational strain, we then take the polar decomposition of $\mathbf{F}$,

$$\mathbf{F} = \mathbf{Q}\tilde{\mathbf{F}}, \tag{3}$$

where $\mathbf{Q}$ is an orthonormal rotation matrix, and $\tilde{\mathbf{F}}$ is symmetric. We then define strain as a linear function of $\tilde{\mathbf{F}}$,

$$\epsilon = \frac{1}{2}\left(\tilde{\mathbf{F}} + \tilde{\mathbf{F}}^T\right) - \mathbf{I}_3, \tag{4}$$

where $\mathbf{I}_3$ is the 3-dimensional identity matrix. Assuming a linear isotropic stress-strain relationship, stress can be computed as

$$\sigma = \lambda \mathrm{Tr}(\epsilon)\,\mathbf{I}_3 + 2\mu\epsilon, \tag{5}$$

where λ and μ are the Lamé material parameters. Elastic forces are given by,

$$\mathbf{f}_i = \frac{1}{3}\mathbf{Q}\sigma\mathbf{n}_i, \tag{6}$$

where $\mathbf{n}_i$ is the area-weighted outward normal of the face opposite node i in reference coordinates. Alternatively, forces may be computed globally as

$$\mathbf{f} = \mathbf{K}(\mathbf{x} - \mathbf{u}), \tag{7}$$

where $\mathbf{K}$ is the global stiffness matrix and the vectors $\mathbf{f}$, $\mathbf{x}$, and $\mathbf{u}$ contain the values for all the nodes of the mesh. The Jacobian of $\mathbf{f}_i$ with respect to node j is the 3×3 matrix,[2]

$$\mathbf{K}_{ij} = -\frac{1}{36v}\mathbf{Q}\left(\lambda\mathbf{n}_i\mathbf{n}_j^T + \mu\left(\mathbf{n}_i \cdot \mathbf{n}_j\right)\mathbf{I}_3 + \mu\mathbf{n}_j\mathbf{n}_i^T\right)\mathbf{Q}^T, \tag{8}$$

where v is the volume of the tetrahedron. In the absence of plasticity or fracture, only the rotation, $\mathbf{Q}$, varies with time, allowing the other terms to be precomputed. In practice, these 3×3 matrices are assembled into a sparse, global stiffness matrix $\mathbf{K}$. For each pair of nodes i and j that share an edge, every tetrahedral element that shares that edge will make a contribution to the 3×3 block in the global stiffness matrix. This is typically implemented as a sum over all elements, with each element computing a new polar decomposition and then, for each of 16 pairs of nodes, performing two 3×3 matrix multiplies and accumulating the result into global stiffness matrix. For Rayleigh damping, the global damping matrix, $\mathbf{D}$, is some scaled combination of $\mathbf{M}$ and $\mathbf{K}$.

3.2 Time Integration

Temporal integration in our implementation is done using linearly implicit Euler. Specifically, we solve the linear system

$$\left(\mathbf{M} - \Delta t^2\mathbf{K} - \Delta t\mathbf{D}\right)\mathbf{v}(t + \Delta t) = \mathbf{M}\mathbf{v}(t) + \Delta t\left(\mathbf{f}_{elc} + \mathbf{f}_{ext}\right), \tag{9}$$

where $\mathbf{M}$ is the lumped nodal mass matrix, $\mathbf{f}_{elc}$ are the elastic forces in Equation (6), and $\mathbf{f}_{ext}$ are external forces. Damping forces do not appear on the right hand side due to cancellation in the derivation. We consider an alternate formulation in Section 4. We advocate solving this linear system with a couple iterations of the conjugate gradient method. As with any optimization, choosing an initial starting point is key to success. We considered three initial guesses for $\mathbf{v}(t + \Delta t)$: $\mathbf{0}$, $\mathbf{v}(t)$, and $\mathbf{v}(t) + \Delta t\mathbf{M}^{-1}\left(\mathbf{f}_{elc} + \mathbf{f}_{ext}\right)$. We achieved the best results by starting the optimization with $\mathbf{v}(t)$.

Positions can then be updated by,

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \Delta t \cdot \mathbf{v}(t + \Delta t). \tag{10}$$

[2]Parker and O'Brien [2009] left out the $1/36v$ term in their version of Equation (8) (Equation (2) in their paper).

Table 1: Table of the simulation time for the examples and initial guesses in milliseconds. All timing is done with the Jacobi preconditioner. The average iteration count for convergence is also supplied. Force, system matrix, and collision are from running to convergence. *These faster computation times are because the simulations quickly go unstable.

Time ms/frame		Max Iterations								
		1	2	5	10	Convergence	Convergence Iterations	Forces	System Matrix	Collision
Bowling	0	0.565	0.581	0.944	1.235	6.689	78	5.465	1.653	227.449
	v(t)+f	0.325	0.364	0.408	1.105	6.614	81	5.422	1.643	226.681
475 Vertices	v(t)	0.515	0.619	0.845	1.196	6.082	75	5.359	1.478	225.102
Snowman	0	0.334	0.466	0.840	1.469	8.991	71	8.336	2.898	-
	v(t)+f	0.211	0.216	0.279	1.465	7.464	58	8.293	2.939	-
508 Vertices	v(t)	0.334	0.468	0.840	1.507	7.771	60	8.331	2.920	-
Cars Spin	0	3.076	4.270	7.924	13.613	74.304	59	87.863	27.107	781.055
	v(t)+f	1.935	2.085	2.520	13.739	74.579	60	87.753	26.710	785.346
4734 Vertices	v(t)	3.143	4.304	7.866	13.745	69.060	55	87.878	23.860	782.972

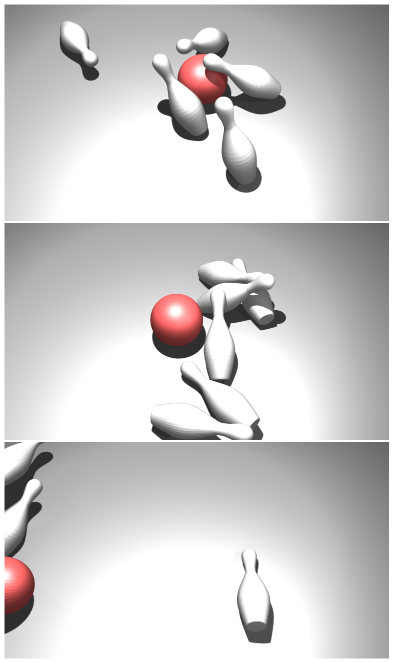

Figure 2: Same frame of the bowling example with 10 conjugate gradient iterations using different initial guesses; 0 (top), $\mathbf{v}(t)$ (middle), and $\mathbf{v}(t) + \Delta t\mathbf{M}^{-1}(\mathbf{f}_{elc} + \mathbf{f}_{ext})$ (bottom)

The initial guess of $\mathbf{0}$ is guaranteed to be stable at every iteration, unlike $\mathbf{v}(t)$, and $\mathbf{v}(t) + \Delta t\mathbf{M}^{-1}(\mathbf{f}_{elc} + \mathbf{f}_{ext})$. However, multiple iterations are required to remove the excess damping. In all our examples, we found that $\mathbf{0}$ required more iterations than $\mathbf{v}(t)$ to reach visually similar results. This can be seen in Figure 3 in the number of iterations it takes to get close to the solution. There were concerns that $\mathbf{v}(t)$ would be unstable with a few iterations from large deceleration during collisions, but this did not occur in practice.

3.3 Polar Decomposition

The most straightforward approach to computing the polar decomposition is to use a "black box" singular value decomposition (SVD) routine, such as `Eigen::JacobiSVD`, to compute $\mathbf{F} = \mathbf{U}\Sigma\mathbf{V}^T$ and then compute $\mathbf{Q} = \mathbf{U}\mathbf{V}^T$ and $\tilde{\mathbf{F}} = \mathbf{Q}^T\mathbf{F}$. However, as suggested by Bridson [2011] a more direct Jacobi iteration approach can be used that consists of iteratively solving 2×2 subproblems using a closed form solution until $\tilde{\mathbf{F}}$ is sufficiently symmetric. Please see the supplementary material for source code for this method. Because Equation (4) essentially symmetrizes $\tilde{\mathbf{F}}$, the decomposition quickly achieves the "sufficiently symmetric" state and, as with conjugate gradients, we have found that a few iterations delivers sufficient accuracy and is substantially faster than off-the-shelf SVD codes.

4 RESULTS AND DISCUSSION

We first analyze the residuals for the different initial guesses and then consider an alternate formulation of our linear system before summarizing the impact of different preconditioners and initial guesses and consider limitations and future work.

Residuals. It is instructive to examine the residuals for each of our initial guesses because the residuals determine the initial search direction for conjugate gradients. Specifically, for residual $\mathbf{r}$ and preconditioner $\mathbf{P}$, the initial solution will be updated by $\alpha\mathbf{P}^{-1}\mathbf{r}$, for some scalar value α. For an initial guess of $\mathbf{0}$, the residual is

$$\mathbf{r} = \mathbf{M}\mathbf{v}(t) + \Delta t\,(\mathbf{f}_{elc} + \mathbf{f}_{ext})\,, \tag{11}$$

which, if $\mathbf{P} = \mathbf{M}$, is essentially the forward Euler direction. For an initial guess of $\mathbf{v}(t)$, we get a residual of

$$\begin{aligned}\mathbf{r} &= \Delta t\,(\mathbf{f}_{elc} + \mathbf{f}_{ext}) + \Delta t^2\mathbf{K}\mathbf{v}(t) + \Delta t\mathbf{D}\mathbf{v}(t)\\ &= \Delta t\left(\mathbf{f}_{elc} + \mathbf{f}_{ext} + \mathbf{f}_{damp}\right) + \Delta t^2\mathbf{K}\mathbf{v}(t),\end{aligned} \tag{12}$$

which again is similar to a forward Euler step with the momentum term removed as it is accounted for in the initial guess. There is the additional $\Delta t^2\mathbf{K}\mathbf{v}(t)$ term, which accounts for elastic forces that would occur if the body continues moving with velocity $\mathbf{v}(t)$.

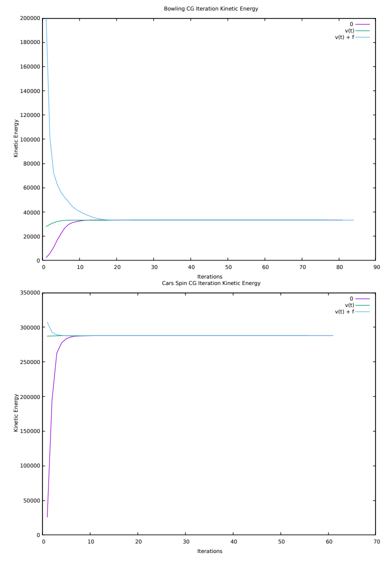

Figure 3: Kinetic energy of a single conjugate gradient solve for the bowling (top) and car spin (bottom) examples.

The residual when choosing $\mathbf{v}(t) + \Delta t\mathbf{M}^{-1}(\mathbf{f}_{elc} + \mathbf{f}_{ext})$ as the initial guess is less intuitive. It is essentially a high pass filter of the forward Euler update.

Alternate Formulation. Baraff and Wikin [1998] used a slightly different formulation of the linear system

$$\left(\mathbf{M} - \Delta t^2\mathbf{K} - \Delta t\mathbf{D}\right)(\mathbf{v}(t+\Delta t) - \mathbf{v}(t)) = \Delta t\left(\mathbf{f}_{elc} + \mathbf{f}_{ext} + \mathbf{f}_{damp}\right) + \Delta t^2\mathbf{K}\mathbf{v}(t), \quad (13)$$

In this case the most straightforward initial guess is $\mathbf{0}$, which resutls in a residual of $\Delta t\left(\mathbf{f}_{elc} + \mathbf{f}_{ext} + \mathbf{f}_{damp}\right) + \Delta t^2\mathbf{K}\mathbf{v}(t)$. Thus this formulation results in identical conjugate gradient iterations as our formulation with an initial guess of $\mathbf{v}(t)$.

Preconditioners and Inital Guess. We also considered different preconditioners. We experimented with the identity $\mathbf{I}$, mass matrix $\mathbf{M}$. and the Jacobi preconditioner $\mathbf{J}$. For all our examples, the identity and mass matrix were roughly equivalent. As shown in Figure 4 for the car sideways collision example, they are indiscernible from each other with all 3 initial guesses. The Jacobi preconditioner typically reached convergence in 1/5th as many iterations as the others and had less variation, most notable in the $\mathbf{v}(t)$ guess in Figure 4. We do not consider the incomplete Cholesky preconditioner in this paper because it needs to be recomputed (or updated) every timestep for corotational finite elements, which is impractical for only a few iterations of conjugate gradients.

Implicit integration removes the need for small time steps and allows us to take large steps. However, by stopping the implicit solve so early, we no longer have this property. Taking the initial guess of the velocity as an example, the time step starts to matter much more with collisions. This initial guess works well because it starts close to the solution, but during collisions that solution will move. As the object is elastic, the solution location doesn't change instantly, but moves there over time. With small time steps, the change per step is small and the early stoppage keeps us close. However, with large time steps we may not get close enough, leading to noticeable errors. All our examples were run with a time step of 1/150, or 5 steps per frame for 30fps. This was sufficient to get usable results with 5-10 iterations. We note that while researchers have long pursued large time steps, Macklin and colleagues [2019] also recently advocated for small time steps in physics simulations.

Table 1 shows the timing results for a couple of the examples. The times are the time in milliseconds per frame to perform the conjugate gradient solve, the force evaluations, building the system matrix, and the collision resolution. The forces, system matrix, and collision columns reported only depend on the mesh resolution. Total simulation time is left out as it depends on the number of iterations run. The average number of iterations to convergence is supplied next to the convergence timing. As each object has its own conjugate gradient solve, we chose the max iteration value as that object is the limiting factor. While $\mathbf{v}(t) + \mathbf{f}$ is included in the table for completion and shows good timing, it is not that beneficial. The bowling and multi examples are not stable with 10 iterations and the cars spinning and snowman example only start being stable around 10 iterations. The initial guess of $\mathbf{0}$ is consistently slightly faster than the velocity, however the examples still have notable artificial damping with 10 iterations.

Limitations and Future Work. This short paper demonstrates that the conjugate gradient method combined with linearly implicit Euler and corotational finite elements may be competitive with existing techniques for real-time animation of soft bodies, but it leaves open many directions for future work, including a more comprehensive comparison to alternative approaches such as projective dynamics [Bouaziz et al. 2014; Liu et al. 2013; Wang 2015]. Our experiments are limited to volumetric soft bodies where the linear system is guaranteed to be positive definite. It would be interesting to explore thin sheets and cloth where the linear systems may have negative Eigenvalues [Liu et al. 2013] that may lead linearly implicit Euler diverge. It would also be interesting to explore the behavior of the conjugate gradient method in a non-linear Newton solver [Chao et al. 2010].

Additional areas of future work include exploring an initial guess between the previous solution and an Euler step, i.e. $\alpha\mathbf{v}(t) + (1-\alpha)\left(\mathbf{v}(t) + \Delta t\mathbf{M}^{-1}(\mathbf{f}_{elc} + \mathbf{f}_{ext})\right)$, and exploring exit criteria beyond iteration count and magnitude of the residual. Finally, our implementation explicitly builds and stores the linear system. It would be interesting to explore a matrix-free implementation; because the matrix is only applied a few times, the cost of building the system may not be sufficiently amortized.

In summary, our results show that conjugate gradients with early termination for corotated finite elements is a viable approach to animating soft bodies. The approach has the advantages of simplicity and extensibility; no precomputation is necessary and features such as fracture and plasticity do not require any special treatment, such as recomputing a matrix factorization.

REFERENCES

[Baraff and Witkin 1998] Baraff and Witkin. 1998. Large Steps in Cloth Simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive*

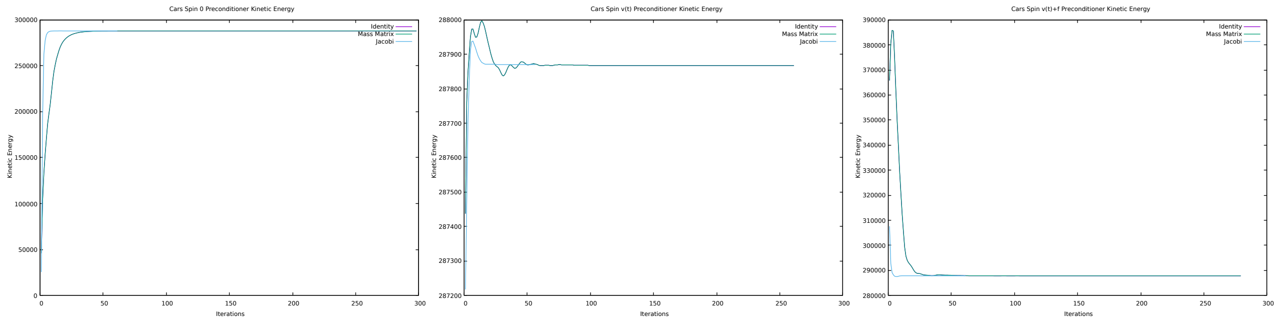

Figure 4: Kinetic energy of a single conjugate gradient solve with different preconditioners for the car spin example with an initial guess of 0 (left), v(t) (middle), and v(t) + f (right). The identity and mass matrix overlap each other on each graph. It is important to note that the y axis does not start at 0 so we can see the differences more clearly.

Techniques (SIGGRAPH '98). ACM, New York, NY, USA, 43–54. https://doi.org/10.1145/280814.280821

[Bouaziz et al. 2014] Bouaziz, Martin, Liu, Kavan, and Pauly. 2014. Projective Dynamics: Fusing Constraint Projections for Fast Simulation. *ACM Trans. Graph.* 33, 4, Article 154 (July 2014), 11 pages. https://doi.org/10.1145/2601097.2601116

[Brandt et al. 2018] Brandt, Eisemann, and Hildebrandt. 2018. Hyper-reduced Projective Dynamics. *ACM Trans. Graph.* 37, 4, Article 80 (July 2018), 13 pages. https://doi.org/10.1145/3197517.3201387

[Bridson 2011] Bridson. 2011. Personal communication.

[Chao et al. 2010] Chao, Pinkall, Sanan, and Schröder. 2010. A Simple Geometric Model for Elastic Deformations. *ACM Trans. Graph.* 29, 4, Article 38 (July 2010), 6 pages. https://doi.org/10.1145/1778765.1778775

[Dinev et al. 2018] Dinev, Liu, Li, Thomaszewski, and Kavan. 2018. FEPR: Fast Energy Projection for Real-time Simulation of Deformable Objects. *ACM Trans. Graph.* 37, 4, Article 79 (July 2018), 12 pages. https://doi.org/10.1145/3197517.3201277

[Hauser et al. 2003] Hauser, Shen, and O'Brien. 2003. Interactive Deformation Using Modal Analysis with Constraints. In *Graphics Interface*. CIPS, Canadian Human-Computer Commnication Society, 247–256. http://graphics.cs.berkeley.edu/papers/Hauser-IDU-2003-06/

[Hecht et al. 2012] Hecht, Lee, Shewchuk, and O'Brien. 2012. Updated Sparse Cholesky Factors for Corotational Elastodynamics. *ACM Trans. Graph.* 31, 5, Article 123 (Sept. 2012), 13 pages. https://doi.org/10.1145/2231816.2231821

[Liu et al. 2013] Liu, Bargteil, O'Brien, and Kavan. 2013. Fast Simulation of Mass-spring Systems. *ACM Trans. Graph.* 32, 6, Article 214 (Nov. 2013), 7 pages. https://doi.org/10.1145/2508363.2508406

[Macklin et al. 2019] Macklin, Storey, Lu, Terdiman, Chentanez, Jeschke, and Müller. 2019. Small Steps in Physics Simulation. In *Proceedings of the 18th Annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '19)*. ACM, New York, NY, USA, Article 2, 7 pages. https://doi.org/10.1145/3309486.3340247

[Müller et al. 2002] Müller, Dorsey, McMillan, Jagnow, and Cutler. 2002. Stable Real-time Deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '02)*. ACM, New York, NY, USA, 49–54. https://doi.org/10.1145/545261.545269

[Müller and Gross 2004] Müller and Gross. 2004. Interactive Virtual Materials. In *Proceedings of Graphics Interface 2004 (GI '04)*. Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 239–246. http://dl.acm.org/citation.cfm?id=1006058.1006087

[Narain et al. 2016] Narain, Overby, and Brown. 2016. ADMM & Supe; Projective Dynamics: Fast Simulation of General Constitutive Models. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '16)*. Eurographics Association, Goslar Germany, Germany, 21–28. http://dl.acm.org/citation.cfm?id=2982818.2982822

[O'Brien and Hodgins 1999] O'Brien and Hodgins. 1999. Graphical Modeling and Animation of Brittle Fracture. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '99)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 137–146. https://doi.org/10.1145/311535.311550

[Parker and O'Brien 2009] Parker and O'Brien. 2009. Real-time Deformation and Fracture in a Game Environment. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '09)*. ACM, New York, NY, USA, 165–175. https://doi.org/10.1145/1599470.1599492

[Shewchuk 1994] Shewchuk. 1994. *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain.* Technical Report. Pittsburgh, PA, USA.

[Wang 2015] Wang. 2015. A Chebyshev Semi-iterative Approach for Accelerating Projective and Position-based Dynamics. *ACM Trans. Graph.* 34, 6, Article 246 (Oct. 2015), 9 pages. https://doi.org/10.1145/2816795.2818063

[Wang and Yang 2016] Wang and Yang. 2016. Descent Methods for Elastic Body Simulation on the GPU. *ACM Trans. Graph.* 35, 6, Article 212 (Nov. 2016), 10 pages. https://doi.org/10.1145/2980179.2980236