Supplement to Optimal r-Adaptive In-Timestep Remeshing for Elastodynamics

JIAHAO WEN, Adobe, USA and University of Southern California, USA
JERNEJ BARBIČ, University of Southern California, USA
DANNY M. KAUFMAN, Adobe, USA

ACM Reference Format:
Jiahao Wen, Jernej Barbič, and Danny M. Kaufman. 2025. Supplement to Optimal r-Adaptive In-Timestep Remeshing for Elastodynamics. *ACM Trans. Graph.* 44, 4 (August 2025), 6 pages. https://doi.org/10.1145/3731204

1 L^2-Projection Assembly

Consider a step of ITR optimization changing the reference mesh from $\mathcal{T}^1(v,e)$ with function space V^1 and basis $\{\phi_i^1 | 1 \le i \le n\}$, to $\mathcal{T}^2(u,e)$ with corresponding function space V^2 and basis $\{\phi_i^2 | 1 \le i \le n\}$. The L^2-projection operator required is then

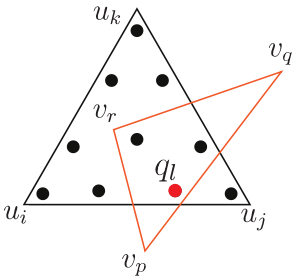
$$\pi : V^1 \to V^2, \tag{1}$$

so that for functions $f^1 \in V^1$ their projection $f^2 = \pi(f^1) \in V^2$ minimizes the L^2 residual $\frac{1}{2}||f^1 - f^2||^2$. For discrete vector fields y^1 defined on $\mathcal{T}^1(v,e)$, the projected vector field y^2 on $\mathcal{T}^2(u,e)$ is given by

$$y^2 = \pi(y^1) = \overline{M}(u)^{-1} A(u,v) y^1, \tag{2}$$

Here $\overline{M}(u)$ is the density-normalized, consistent mass matrix on the mesh $\mathcal{T}^2(u,e)$, and $A(u,v)$ is the transfer matrix between bases so that submatrix blocks $A_{i,j} = \int_\Omega \phi_j^1 \phi_i^2 dV \otimes I_d$.

We apply quadratures to compute the inner products of the two meshes', $\mathcal{T}^1$ and $\mathcal{T}^2$, shape functions. Here we demonstrate the assembly of the transfer matrix using a 2D triangulation as an example. Assembly for tetrahedral meshes in 3D follows directly. For each triangle $\triangle(u_i,u_j,u_k)$ in the modified reference mesh, $\mathcal{T}^2(u,e)$, we apply quadrature sampling with m quadrature points $\{q_l | 1 \le l \le m\}$ and corresponding quadrature weights $\{w_l | 1 \le l \le m\}$.



We construct an AABB-based BVH on $\mathcal{T}^1(v,e)$. For each quadrature point q_l, we use this BVH to accelerate the look up of which triangle $\triangle(v_p,v_q,v_r)$ in the original mesh, $\mathcal{T}^1$, contains it. We then compute the barycentric coordinates $\phi^1_{p,q,r}$ of q_l in $\triangle(v_p,v_q,v_r)$ and the barycentric coordinates $\phi^2_{i,j,k}$ of q_l in $\triangle(u_i,u_j,u_k)$. The contribution of the quadrature point q_l to the transfer matrix $A(u,v)$ is then

$$w_l W_{ijk} \sum_{\alpha=p,q,r} \sum_{\beta=i,j,k} \phi^1_\alpha \phi^2_\beta, \tag{3}$$

where W_{ijk} gives the triangle area of $\triangle(u_i,u_j,u_k)$.

2 Computing the L^2-Projection Derivatives

With assembly covered above we now focus on our computation of projection derivatives in terms of the operator,

$$b(y,x,u,v) = \rho\, x^T A(u,v)\, y. \tag{4}$$

As a concrete example of why we focus on this operator recall our inertial energy is

$$K(x,\tilde{x}^t,\mathcal{T}) = \frac{1}{2} x^T M(\mathcal{T}) x - x^T M(\mathcal{T}) \pi_t(\tilde{x}^t,\mathcal{T}). \tag{5}$$

Substituting the projection, $\pi_t(\tilde{x}^t,\mathcal{T})$, via Equation (2), we then have (with the assumption of constant, per-body density),

$$\begin{aligned} K(x,\tilde{x}^t,\mathcal{T}) &= \frac{1}{2} x^T M(\mathcal{T}) x - x^T M(\mathcal{T}) \overline{M}(\mathcal{T})^{-1} A_t \tilde{x}^t \\ &= \frac{1}{2} x^T M(\mathcal{T}) x - \rho\, x^T A_t \tilde{x}^t. \end{aligned} \tag{6}$$

More generally, for optimal ITR, we need only consider computing the derivatives of $b = \rho x^T A y$, where y is an arbitrary constant vector, to account for projection terms. In the following, to simplify, we drop the constant ρ term, and use the same quadrature application (and notation) as in the last section to evaluate b and its derivatives.

The contribution of each quadrature point q_l to b is then

$$b_l = w_l W_{ijk} \sum_{\alpha=p,q,r} \sum_{\beta=i,j,k} \phi^1_\alpha \phi^2_\beta x^T_\beta y_\alpha. \tag{7}$$

Triangle areas W_{ijk} are then functions of $u_{i,j,k}$, while the quadrature weights, w_l, are constant. As q_l is interpolated from u_i,u_j,u_k, the corresponding shape functions, ϕ^1_α, are also a linear function of $u_{i,j,k}$. In turn, shape functions ϕ^2_β are constant values predefined by the quadrature pattern, while the input vector y_α is also constant.

Authors' Contact Information: Jiahao Wen, Adobe, USA and University of Southern California, USA, jiahaow@usc.edu; Jernej Barbič, University of Southern California, USA, jnb@usc.edu; Danny M. Kaufman, Adobe, USA, dannykaufman@gmail.com.

ACM 1557-7368/2025/8-ART
https://doi.org/10.1145/3731204

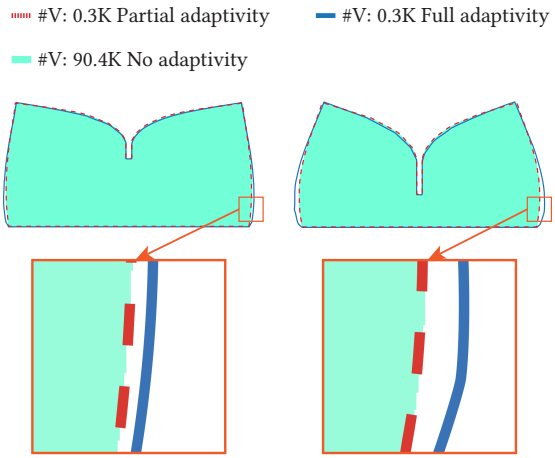


Fig. 2. **Supplement to the deep elastic punch.** See our detailed analysis in the main paper.

Gradient terms are then

$$\frac{\partial b_l}{\partial u_\gamma} = w_l \frac{\partial W_{ijk}}{\partial u_\gamma} \Big(\sum_{\alpha=p,q,r} \sum_{\beta=i,j,k} \phi^1_\alpha \phi^2_\beta x^T_\beta y_\alpha \Big) + w_l W_{ijk} \sum_{\alpha=p,q,r} \sum_{\beta=i,j,k} \frac{\partial \phi^1_\alpha}{\partial u_\gamma} \phi^2_\beta x^T_\beta y_\alpha, \tag{8}$$

$$\frac{\partial b_l}{\partial x_\gamma} = w_l W_{ijk} \sum_{\alpha=p,q,r} \phi^1_\alpha \phi^2_\gamma y_\alpha,$$

and corresponding Hessian terms are

$$\begin{aligned}
\frac{\partial^2 b_l}{\partial u_\gamma \partial u_\delta} &= w_l \frac{\partial^2 W_{ijk}}{\partial u_\gamma \partial u_\delta} \Big(\sum_{\alpha=p,q,r} \sum_{\beta=i,j,k} \phi^1_\alpha \phi^2_\beta x^T_\beta y_\alpha \Big) \\
&+ w_l \frac{\partial W_{ijk}}{\partial u_\gamma} \Big(\sum_{\alpha=p,q,r} \sum_{\beta=i,j,k} \frac{\partial \phi^1_\alpha}{\partial u_\delta} \phi^2_\beta x^T_\beta y_\alpha \Big)^T \\
&+ w_l \frac{\partial W_{ijk}}{\partial u_\delta} \Big(\sum_{\alpha=p,q,r} \sum_{\beta=i,j,k} \frac{\partial \phi^1_\alpha}{\partial u_\gamma} \phi^2_\beta x^T_\beta y_\alpha \Big)^T \\
&+ w_l W_{ijk} \sum_{\alpha=p,q,r} \sum_{\beta=i,j,k} \frac{\partial^2 \phi^1_\alpha}{\partial u_\gamma \partial u_\delta} \phi^2_\beta x^T_\beta y_\alpha, \quad \to 0 \\
\frac{\partial^2 b_l}{\partial u_\gamma \partial x_\delta} &= w_l \frac{\partial W_{ijk}}{\partial u_\gamma} \Big(\sum_{\alpha=p,q,r} \phi^1_\alpha \phi^2_\delta y_\alpha \Big)^T + w_l W_{ijk} \sum_{\alpha=p,q,r} \frac{\partial \phi^1_\alpha}{\partial u_\gamma} \phi^2_\delta y^T_\alpha \\
\frac{\partial^2 b_l}{\partial x_\gamma \partial x_\delta} &= 0.
\end{aligned} \tag{9}$$

References

J Mosler and M Ortiz. 2006. On the numerical implementation of variational arbitrary Lagrangian–Eulerian (VALE) formulations. *Internat. J. Numer. Methods Engrg.* 67, 9 (2006).

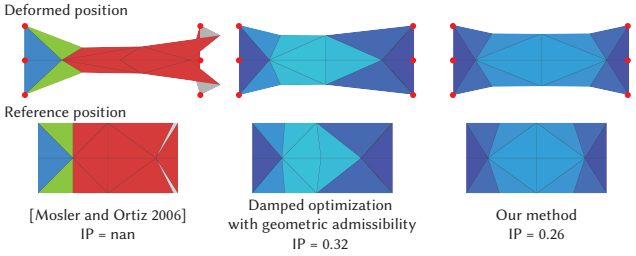


Fig. 1. Using the same example as in Figure 8 of the main paper we compare Mosler and Ortiz's [2006] proposed damped Newton solve (left column) of the unconstrained incremental potential with, for reference, our optimal r-ITR ("our method", right column) solution. As discussed in the main paper, even in this simple 2D example, Mosler and Ortiz's unconstrained solution reaches degenerate meshes with inverted elements. In the middle column we experiment by *augmenting* Mosler and Ortiz [2006] with an additional admissibility constraint that disallows inverted elements. Here, we see the Mosler and Ortiz solution improves somewhat, but now, as expected, with a degenerate mesh that has close-to-collapsed elements instead of inverted ones.

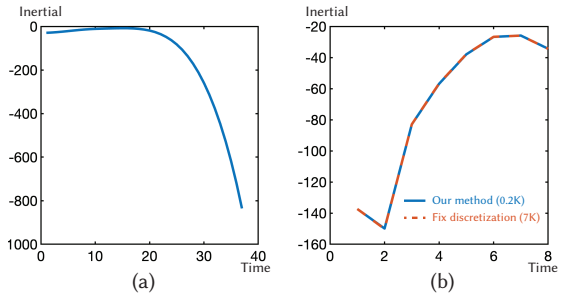


Fig. 3. ITR solutions avoid trajectory jumps and jitters, and maintain physically consistent kinetic energy by (1) solving for r-adaptive updates inside each timestep, and (2) by ensuring consistent joint remapping of velocities and mass matrix on each remeshing update. Optimal ITR trajectories are then smooth (with correspondingly smooth kinetic energy evolution) where dynamics are expected to be so; see (a) where we plot the kinetic energy over time of an optimal r-ITR simulation of the 3D drop-through example. Of course optimal r-ITR will then exhibit sharper changes where physically appropriate, as during impacts: see (b) where we overlay the kinetic energy plot of a coarse optimal r-ITR simulation of the high-speed impact example with a much (35X) finer fixed mesh simulation of the same.

Table 1. **Timing breakdown.** Here we breakdown timing statistics for individual examples. "Evaluation" gives total time in evaluating energy gradients and Hessians that form the linear system per Newton iterate. "Linear solve" gives total solve time for linear system solves. "Line search" is the corresponding time used to perform line search in finding optimal meshes; note this includes inner $\text{argmin}_x E_t$ solves involved. "L2-assembly" is the time to compute the transfer matrix for L^2-projections. "L2-derivative" is the time to compute the L^2-projection derivatives in the inertial energy.

Example	#V	#E	Linear solve (s)	Line search (s)	Evaluation (s)	L2-assembly (s)	L2-derivative (s)	Timestep	Timing per timestep (s)
High-speed impact	218	392	0.029	2.479	1.481	0.788	1.194	0.010	4.057
3D drop through	234	588	0.010	2.846	2.006	2.127	1.777	0.010	5.000
Gingerbread	694	1,266	0.017	0.841	0.524	0.351	0.424	0.010	1.483
Masticator	738	2,508	0.042	6.495	16.130	7.524	15.205	0.050	25.100
Chicken	10,218	31,555	0.237	49.845	46.919	29.791	41.897	0.010	118.169
Jelly	11,139	46,347	1.064	134.651	194.437	132.345	173.113	0.010	347.006

Table 2. Simulation parameters.

Example	Young's modulus	Poisson's ratio	E_tol	R_tol	timestep	frictional coefficient
Jelly	2.00E+06	0.1	1.00E-04	1.00E-04	1.00E-02	0.3
Chicken	4.00E+04	0.4	1.00E-04	1.00E-03	1.00E-02	0.15
Stamps	2.00E+05	0.3	1.00E-04	1.00E-04	1.00E-02	0
Masticator	1.00E+04	0.4	1.00E-04	5.00E-04	5.00E-02	0
High-speed impact	2.00E+05	0.3	1.00E-04	1.00E-04	1.00E-02	0
Gingerbread	2.00E+05	0.2	1.00E-04	1.00E-04	1.00E-02	0.1
Bending beam test	2.00E+06	0.3	1.00E-04	1.00E-04	static	0
3D drop through	2.00E+05	0.25	1.00E-04	1.00E-04	1.00E-02	0
2D static drop through	4.00E+04	0.05	1.00E-04	1.00E-04	static	0
Deep elastic punch	2.00E+05	0.3	1.00E-04	1.00E-04	static	0
L2-projection	2.00E+04	0.3	1.00E-04	1.00E-04	1.00E-02	0.2

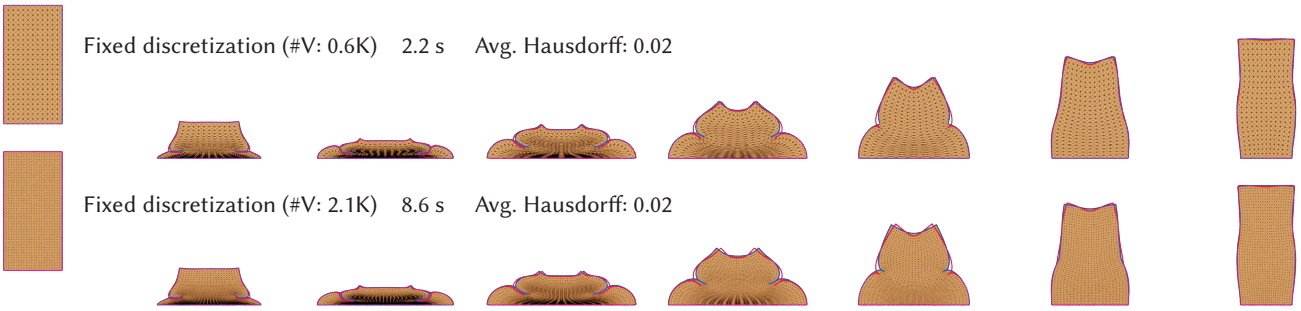


Fig. 4. **Additional fixed discretization resolutions for reference for the *High-speed impact and transients* example.**

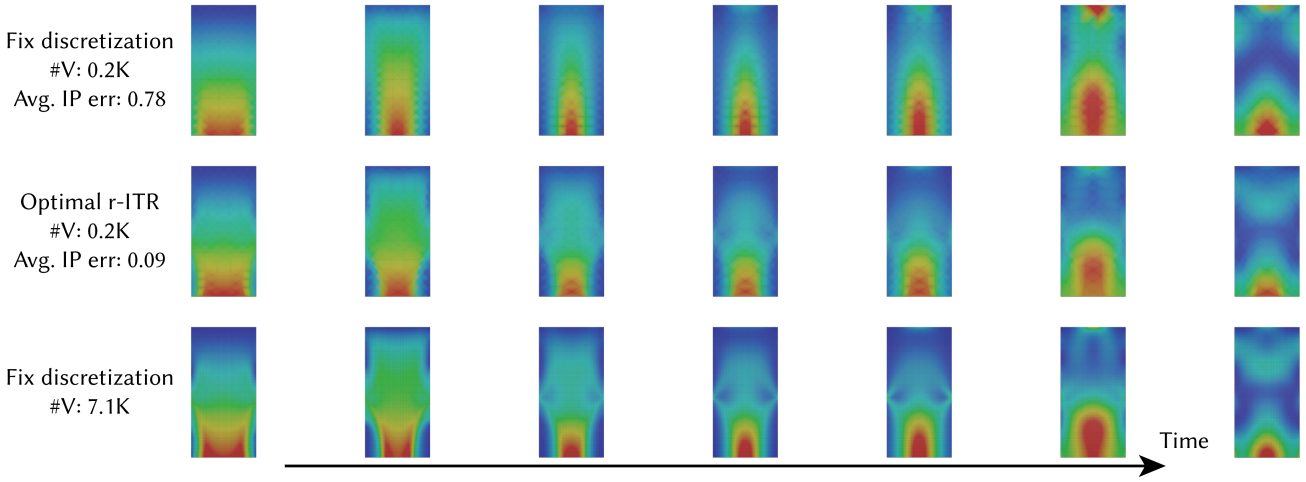


Fig. 5. **Additional optimal r-ITR and fixed discretization reference solutions for the *High-speed impact and transients* example.**

Table 3. Performance statistics.

Example	#Vertices	#Elements	Timing per timestep (s)	Memory (MB)
Masticator (Optimal r-ITR)	738	2,508	25.100	679
Masticator (Ferguson et al. 2023)	484–21273	1740–97667	1,830.900	2,619
Masticator (Fixed mesh)	71,057	384,042	210.830	5,111
3D drop through (Optimal r-ITR)	234	588	5.000	255
3D drop through (Fixed mesh)	234	588	0.030	94
3D drop through (Fixed mesh)	1,445	5,447	0.582	273
3D drop through (Fixed mesh)	6,416	26,191	5.776	500
3D drop through (Fixed mesh)	27,296	115,719	46.494	1,389
3D drop through (Fixed mesh)	113,421	489,726	556.798	4,711
2D static drop through (Optimal r-ITR)	117	196	2.158	74
2D static drop through (Optimal r-ITR)	429	784	7.065	122
2D static drop through (Optimal r-ITR)	1,641	3,136	67.009	284
2D static drop through (Optimal r-ITR)	6,417	12,544	2,353.040	1,659
2D static drop through (Optimal r-ITR)	25,377	50,176	19,620.626	5,401
2D static drop through (Fixed mesh)	117	196	0.072	68
2D static drop through (Fixed mesh)	429	784	0.271	72
2D static drop through (Fixed mesh)	1,641	3,136	3.057	91
2D static drop through (Fixed mesh)	6,417	12,544	47.177	190
2D static drop through (Fixed mesh)	25,377	50,176	415.183	440
2D static drop through (Fixed mesh)	100,957	200,704	4,486.742	747
2D static drop through (Fixed mesh)	402,561	802,816	57,018.258	2,336

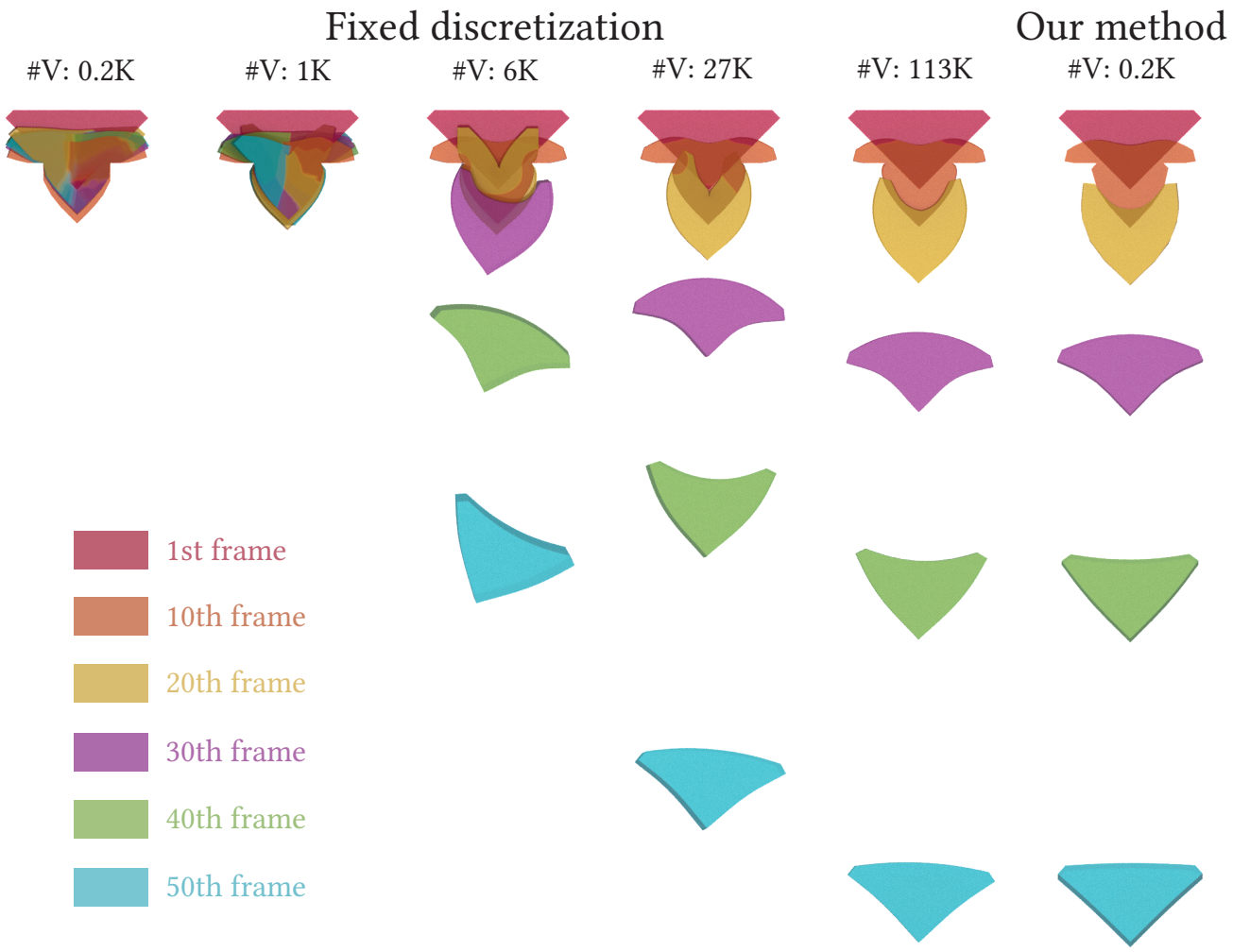


Fig. 6. **Supplement to the *Dynamic drop through* example.** We compare overlaid frames of the successively finer fixed-discretization simulations with our optimal r-ITR's coarse simulation in the drop through example. Optimal ITR keeps comparable height with the finest fixed-mesh simulation along the trajectory.

Table 4. Performance statistics.

Example	#Vertices	#Elements	Timing per timestep (s)	Memory (MB)
Deep elastic punch1 (Optimal r-ITR)	81	128	0.129	87
Deep elastic punch1 (Optimal r-ITR)	285	512	0.378	149
Deep elastic punch1 (Optimal r-ITR)	617	1,152	0.666	194
Deep elastic punch1 (Optimal r-ITR)	1,077	2,048	1.053	189
Deep elastic punch1 (Optimal r-ITR)	2,580	5,000	2.656	470
Deep elastic punch1 (Optimal r-ITR)	6,525	12,800	4.921	1,752
Deep elastic punch1 (Optimal r-ITR)	10,155	20,000	8.436	3,665
Deep elastic punch1 (Optimal r-ITR)	12,270	24,200	16.211	5,052
Deep elastic punch1 (Optimal r-ITR)	13,172	25,992	17.468	5,171
Deep elastic punch1 (Fixed mesh)	81	128	0.004	82
Deep elastic punch1 (Fixed mesh)	285	512	0.012	134
Deep elastic punch1 (Fixed mesh)	617	1,152	0.042	153
Deep elastic punch1 (Fixed mesh)	1,077	2,048	0.057	162
Deep elastic punch1 (Fixed mesh)	2,580	5,000	0.155	238
Deep elastic punch1 (Fixed mesh)	6,525	12,800	0.331	452
Deep elastic punch1 (Fixed mesh)	10,155	20,000	0.650	723
Deep elastic punch1 (Fixed mesh)	17,100	33,800	1.744	1,103
Deep elastic punch1 (Fixed mesh)	40,305	80,000	3.603	2,352
Deep elastic punch1 (Fixed mesh)	62,880	125,000	7.566	3,590
Deep elastic punch1 (Fixed mesh)	90,455	180,000	19.343	4,681
Deep elastic punch2 (Optimal r-ITR)	81	128	0.136	86
Deep elastic punch2 (Optimal r-ITR)	285	512	0.483	150
Deep elastic punch2 (Optimal r-ITR)	617	1,152	0.863	197
Deep elastic punch2 (Optimal r-ITR)	1,077	2,048	1.475	185
Deep elastic punch2 (Optimal r-ITR)	2,580	5,000	4.539	468
Deep elastic punch2 (Optimal r-ITR)	6,525	12,800	11.225	1,784
Deep elastic punch2 (Optimal r-ITR)	10,155	20,000	23.710	3,672
Deep elastic punch2 (Optimal r-ITR)	12,270	24,200	35.415	4,961
Deep elastic punch2 (Fixed mesh)	81	128	0.004	85
Deep elastic punch2 (Fixed mesh)	285	512	0.013	136
Deep elastic punch2 (Fixed mesh)	617	1,152	0.047	156
Deep elastic punch2 (Fixed mesh)	1,077	2,048	0.061	164
Deep elastic punch2 (Fixed mesh)	2,580	5,000	0.157	244
Deep elastic punch2 (Fixed mesh)	6,525	12,800	0.454	450
Deep elastic punch2 (Fixed mesh)	17,100	33,800	1.670	1,099
Deep elastic punch2 (Fixed mesh)	40,305	80,000	6.811	2,345
Deep elastic punch2 (Fixed mesh)	62,880	125,000	11.176	3,504
Deep elastic punch2 (Fixed mesh)	90,455	180,000	29.146	4,566