

# Fine Wrinkling on Coarsely-Meshed Thin Shells

ZHEN CHEN, University of Texas at Austin, USA

HSIAO-YU CHEN, University of Texas at Austin, USA

DANNY M. KAUFMAN, Adobe Research, USA

MÉLINA SKOURAS, Université Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, France

ETIENNE VOUGA, University of Texas at Austin, USA

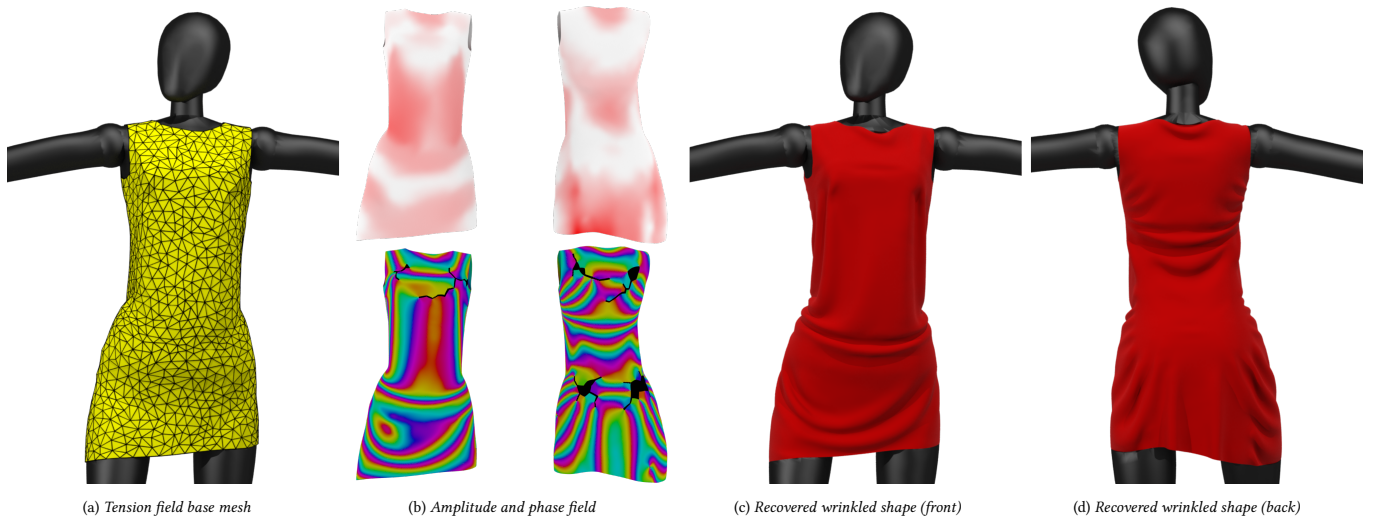


Fig. 1. Overview of our pipeline for predicting the wrinkled equilibrium shape of a thin shell (in this case, a cloth dress draped on a mannequin). We approximate the coarse shape of the draped cloth using tension field theory (a), in which material forces do not resist compression. We then augment this *base mesh*, which can be very coarse (around one thousand vertices), with wrinkles. We formulate the elastic energy of the shell in terms of an amplitude (b, *top*) and phase field (b, *bottom*) over the base mesh, which together characterize the geometry of the wrinkles, and solve for these fields globally over the mesh. Our method recovers complex wrinkle patterns with nontrivial geometry and topology (c, d), including wrinkles with wavelength much smaller than the resolution of the base mesh.

We propose a new model and algorithm to capture the high-definition statics of thin shells via coarse meshes. This model predicts global, fine-scale wrinkling at frequencies much higher than the resolution of the coarse mesh; moreover, it is grounded in the geometric analysis of elasticity, and does not require manual guidance, a corpus of training examples, nor tuning of ad-hoc parameters. We first approximate the coarse shape of the shell using tension field theory, in which material forces do not resist compression. We then augment this base mesh with wrinkles, parameterized by an amplitude and phase field that we solve for over the base mesh, which together characterize

the geometry of the wrinkles. We validate our approach against both physical experiments and numerical simulations, and show that our algorithm produces wrinkles qualitatively similar to those predicted by traditional shell solvers requiring orders of magnitude more degrees of freedom.

CCS Concepts: • **Computing methodologies** → **Physical simulation**; **Mesh models**.

Additional Key Words and Phrases: Physical simulation, tension field theory, wrinkle simulation, cloth simulation, mechanics of thin shells

## ACM Reference Format:

Zhen Chen, Hsiao-yu Chen, Danny M. Kaufman, Mélina Skouras, and Etienne Vouga. 2021. Fine Wrinkling on Coarsely-Meshed Thin Shells. *ACM Trans. Graph.* 1, 1, Article 1 (January 2021), 32 pages. <https://doi.org/10.1145/3462758>

## 1 INTRODUCTION

Complex and high-frequency wrinkling patterns give thin-sheet and membrane materials, like cloth and plastic film, their characteristic fine details. Yet the rich diversity of wrinkling behaviors observable in these everyday materials also pose a significant modeling and simulation challenge: failure to represent the material using

Authors' addresses: Zhen Chen, University of Texas at Austin, USA, [zchen96@cs.utexas.edu](mailto:zchen96@cs.utexas.edu); Hsiao-yu Chen, University of Texas at Austin, USA, [hsiaoyu@ices.utexas.edu](mailto:hsiaoyu@ices.utexas.edu); Danny M. Kaufman, Adobe Research, USA, [dannykaufman@gmail.com](mailto:dannykaufman@gmail.com); Mélina Skouras, Université Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, France, [melina.skouras@inria.fr](mailto:melina.skouras@inria.fr); Etienne Vouga, University of Texas at Austin, USA, [evouga@cs.utexas.edu](mailto:evouga@cs.utexas.edu).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).

© 2021 Copyright held by the owner/author(s). Publication rights licensed to ACM.  
0730-0301/2021/1-ART1 \$15.00  
<https://doi.org/10.1145/3462758>

sufficient degrees of freedom to adequately sample the surface's highest-frequency wrinkle features leads to noisy, aliased results, or else overly-smoothed surfaces whose fine wrinkles are missing altogether. Thus computational meshes with hundreds of thousands of vertices can be required in computer animation or garment modeling applications in order to capture accurate and/or expressive wrinkles.

Towards the efficient and accurate simulation of wrinkling we propose a new model and algorithm to capture the high-definition statics of thin shells via coarse meshes. Our goal is to model the complex and fine wrinkles that arise in the interplay between tension and compression while utilizing just a small number of degrees of freedom. For example, in the dress shown in Figure 1 we simulate with 1.4k degrees of freedom (compare to a full resolution simulation result using 40k vertices in Figure 17). The key idea of our approach is to split the kinematics of wrinkled shells into a *base mesh*, representing the coarse envelope of the shell without its fine wrinkles, and a *wrinkle field* parameterized over the base mesh, encoding the amplitude and wavelength of high-frequency wrinkles. In the first half of our paper, we formulate a reduced-order model of shell statics in terms of these degrees of freedom. In the second, we propose a concrete algorithm for computing the wrinkled static shape of tension-dominated thin materials, such as draped garments or inflated balloons, based on the choice of *tension field theory* for computing the base mesh shape.

Our model predicts global, fine-scale wrinkling at frequencies much higher than the resolution of the base mesh and, in contrast to previous heuristic approaches for wrinkle augmentation, is grounded in the geometric analysis of elasticity, and does not require manual guidance, a corpus of training examples, nor tuning of ad-hoc parameters. We validate our approach against both physical experiments and numerical simulations, show that our algorithm generates both well-known experimental results and simulations with wrinkle quality comparable to those obtained by classical cloth solvers utilizing orders of magnitude more degrees of freedom.

**Wrinkle Fields.** Augmenting a coarse simulation with additional high-frequency detail, via techniques such as normal or displacement mapping (either crafted by artists or learned from data [Löhner et al. 2018]), is a long-standing and powerful strategy. We adopt the approach of several prior works in physics and computer graphics which parameterize wrinkles as spatially-varying amplitude and phase fields. To solve for these wrinkle fields and add fine wrinkles to a base mesh, previous methods explore several ideas based on *local* analyses of its deformation, either by assuming that the base mesh has a very simple geometry, so that wrinkle behavior can be predicted analytically; by restricting to materials like skin where a volumetric substrate drives local wrinkling; or by procedurally modeling wrinkles based on heuristics, user guidance, or data-driven models. While such local models can capture the wrinkling of pinched skin or tight-fitting spandex, where the wrinkle frequency and amplitude are indeed determined by local rather than long-range interactions, they are inapplicable to loose-fitting clothing (such as the dress in Figure 1) or draped cloth. In contrast, we develop a principled *global* model of wrinkle fields, grounded in the physics of thin shells.

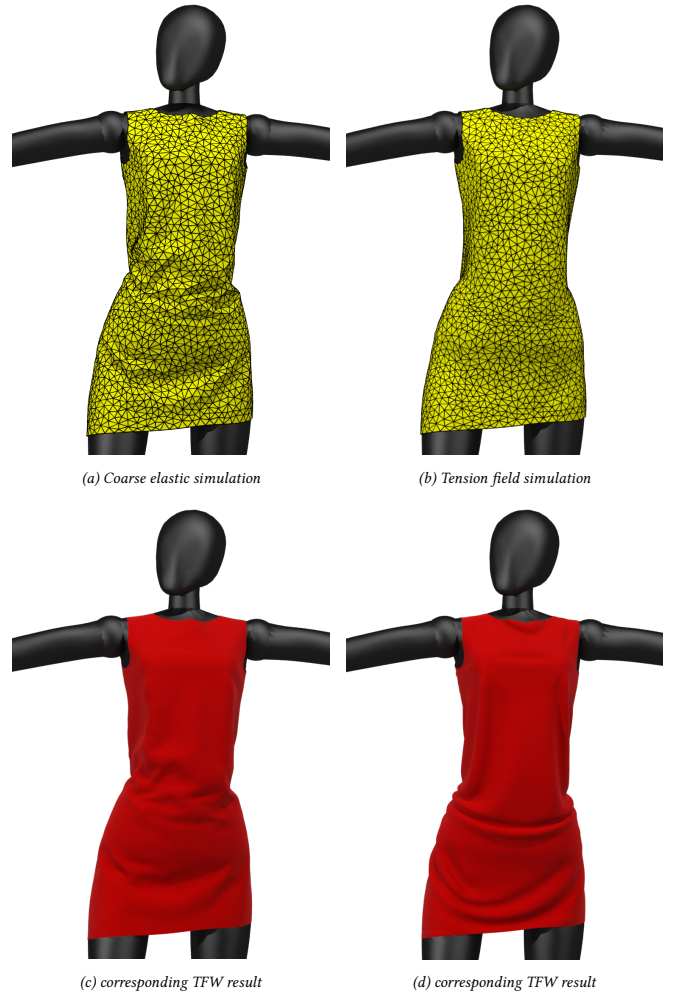


Fig. 2. Augmenting a coarse elastic simulation (*left*) and tension-field simulation (*right*) of a dress with wrinkles. Note that using a coarse elastic simulation yields poor results: since a coarse mesh cannot represent fine wrinkling, the simulation produces an *aliased* result with incorrect, coarse wrinkling instead. Adding additional high-frequency wrinkles to this base mesh is not useful. Instead, we propose using a tension-field simulation of the shell as the base mesh; the *TFT* solution is devoid of wrinkles, and becomes a blank canvas on which we can solve for a high-quality wrinkle field.

**Problem Scope.** We apply our reduced-order wrinkle model to the problem of simulating the static shape of thin shells under a mix of tension and compression, subject to boundary conditions. Here we focus exclusively on this statics problem with potential applications to virtual try-on, garment design, draping, and modeling. We do not consider dynamics, although see Section 7 for discussion of how the theory we develop might be applied to dynamics. We solve the statics problem by first computing the base mesh shape, and then solving for the wrinkle orientation, frequency, and amplitude.



*Choice of Base Mesh.* Although a coarse-resolution simulation of the shell may be the most obvious choice of base mesh, this choice is flawed (see Figure 2). If a shell would wrinkle at a frequency higher than can be resolved by the tessellation, the coarse-resolution simulated shell will buckle, but at an aliased frequency much coarser than the physically-correct wrinkle features. Augmenting such a base mesh with plausible wrinkles would thus require *both* removing the existing, spurious coarse buckling, and then inserting new high-frequency wrinkles. Instead, we propose computing a base mesh free of any wrinkles using *tension field theory* (TFT); this TFT base mesh then serves as a blank canvas upon which we can optimize for a wrinkle field encoding fine details at the correct frequency and amplitude.

*Tension Field Theory.* The key insight of tension field theory [Pipkin 1986; Steigmann 1990] is that tension-dominated thin shell deformation can be understood at two independent scales: the coarse structure and the fine wrinkles. Consider, for instance, manipulating a piece of cloth: the cloth strongly resists *extension*, but allows *compression* with almost no resistance. When compressed, the cloth buckles into a wrinkled shape, at very low energy cost since the bending stiffness of real materials is orders of magnitude weaker than the stretching stiffness. The dominant forces determining the *coarse* structure of cloth that is being tugged or draped are therefore internal tension forces, gravity and other external loads, and bending; the internal compression forces and the buckling and wrinkling they induce play a negligible role in the cloth’s overall shape.

Despite their relative unimportance in determining coarse shape, compression forces are the chief source of numerical difficulty when simulating shell statics. Not only does resolving the wrinkles induced by compression require finely-tessellated elements, but the elastic energy of compression is non-convex (since there is symmetry in whether a small, compressed strip of cloth will buckle upwards or downwards; there is also phase-shift symmetry in the wrinkle pattern on a larger cloth patch). In cases where simulating detailed wrinkling is not required, such as when designing inflatable balloons [Skouras et al. 2014], there has been great profit in neglecting the compressive forces altogether: the resulting TFT simulation minimizes a convex elastic energy which accurately predicts a surface’s coarse-scale shape even when the simulation mesh has few degrees of freedom. For tension-dominated problems, such as simulating cloth drapes or deformation of pressurized chambers, TFT is thus ideal for generating an approximate coarse base mesh for applying our wrinkle fields (constructed in Section 3).

*Contributions.* To summarize, our core contributions include:

- a new formulation of thin shell kinematics that splits degrees of freedom between coarse-scale deformation of the surface, and high-frequency wrinkling, expressed as functions on the coarse surface. We derive an energy model for the shell in terms of these degrees of freedom (Section 3);
- we propose an algorithm which uses the above energy model to solve for the static shape of cloth and other thin materials, when subject to loads that induce a mix of tension and compression (Section 4). At its heart, this algorithm first solves for the coarse shape using TFT simulation, and then, given

the coarse mesh, solves a sequence of quadratic programs to compute the wrinkle field parameters;

- we evaluate our model and algorithm on a variety of test cases, including experiments drawn from the physics literature, and simulations of garment drapes and inflatable structures (Section 6). We show qualitative agreement between our results and those of both established experiments and full degree-of-freedom shell solvers, even when our method is discretized very coarsely, e.g., using  $\approx 1000$  degrees of freedom; and
- although low-level performance optimization is not our focus here, we show that our method’s advantage of working on much coarser meshes translates into speedups (up to one or two orders of magnitude) when compared to a baseline, optimized Newton-type solver for shell statics (Section 6).

We find that coarse base meshes, together with wrinkle fields, are a powerful representation for simulating cloth and other thin materials with complex, highly-detailed, high-frequency wrinkle-like features, and yield striking results in comparison to simulations using classic bending and stretching elements. We hope our model will serve as a foundation for further research into simulation using wrinkle-field kinematics, not only for forward problems, but also for inverse design problems, where the wrinkle amplitude and direction are more natural and semantically meaningful deformation degrees of freedom than traditional vertex displacements.

## 2 RELATED WORK

Thin shell simulation has long been a research focus in both computational mechanics and computer graphics. Considerable effort has focused on improving computational efficiency of generic cloth and shell solvers. This work was pioneered by Baraff and Witkin’s [1998] application of implicit time integration to accelerate cloth simulation. Subsequent research proposes many improvements including implicit-explicit methods [Boxerman and Ascher 2004], adaptive remeshing [Grinspun et al. 2002; Li et al. 2018; Narain et al. 2013, 2012], distributed memory parallelism [Selle et al. 2009], position-based dynamics [Müller et al. 2007], subdivision thin shell element methods [Vetter et al. 2014], multi-grid methods [Tamstorf et al. 2015], and various approaches to incorporating yarn-level dynamics [Kaldor et al. 2008], such as by homogenization [Sperl et al. 2020] or enrichment of a triangle mesh by yarn patches [Casafranca et al. 2020].

*Theoretical Analysis of Wrinkles.* The interplay of thin shell mechanics and wrinkling has been significantly studied in the physics community. Cerda and Mahadevan [2003] derive a scaling law that relates the wrinkle wavelength to the material parameters of a stretched elastic thin sheet; this experiment was analyzed systematically in follow-on computational work [Healey et al. 2013; Li and Healey 2016; Wang et al. 2018]. The Cerda and Mahadevan model was later extended to more complex deformations and geometries [Aharoni et al. 2017; Paulsen et al. 2016]. The work of Paulsen et al. [2016] is particularly notable as it accounts for the role of surface curvature on wrinkling; their extension can be used to predict the wavelength of wrinkles on simple, rotationally-symmetric 3D geometries like cylinders and hemispheres. However, their analysis assumes that the wrinkle direction is known in advance and

that wavelength is constant in this direction, which is not true for everyday, complex geometries, for example in draped dresses or pants.

*Physically-Inspired Wrinkle Simulation.* Given the importance of fine wrinkles to the visual quality of cloth simulation, several methods have studied augmenting simulations to better reproduce high-quality wrinkles, either by modifying the solve itself, or adding wrinkles as a post-process. Bergou et al. [2007] use constrained Lagrangian mechanics to force a high resolution simulation to track the coarse motion of an art-directed low-resolution target surface, thereby enhancing the coarse motion with fine-scale details. Remillard and Kry [2013] apply a similar idea to simulation of skin, by using a sparse set of constraints to couple the motion of outer skin layers to the underlying volumetric substrate. In a similar vein, Wrinkle Meshes [Müller and Chentanez 2010] computes fine wrinkles by simulating high-resolution patches constrained to an initial, low-resolution simulation. Although both methods generate a high-resolution wrinkled surface whose coarse shape matches a coarse target, these tracking-based methods still require simulating wrinkles on a high-resolution mesh, in contrast to our approach based on coarse wrinkle-field kinematics. Several post-processing methods have been proposed that add dynamic wrinkles based on analyzing the strain tensor after a coarse-scale simulation [Gillette et al. 2015; Rohmer et al. 2010]; although real-time, these methods rely on user guidance to choose the proper wrinkle size, rather than inferring the correct wrinkles from the cloth physics.

Zuenko et al. [2019] predict the wrinkling of human skin, and other materials consisting of a stiff film coupled to a soft substrate. Similar to our approach, Zuenko et al. solve for an amplitude and phase field discretized on a surface mesh. Different from us, however, Zuenko et al. compute wrinkle frequency based on local phenomenological laws drawn from the physics literature that relate wrinkle frequency to the film-to-substrate shear modulus ratio. Their method is thus only applicable to wrinkled skin or spandex, and not to wrinkling of loose-fitting cloth and other shells which are not bonded to any substrate.

In case of no substrate, one way to extend Zuenko et al.’s approach is to use an ad-hoc value of film-to-substrate shear modulus ratio to control the wrinkle frequency; this appears to be how Zuenko et al. simulated Figures 8a,b in their paper, for instance (showing wrinkling on a toroidal balloon and a gold sheet draped on a sphere). As in the techniques above that add dynamics as a postprocess, the downside of this idea is that one now needs to manually tune this parameter, rather than allowing the physics to create the wrinkles automatically. As an alternative, for shells with no substrate, Zuenko et al. also extended their approach by using the scaling law of Vandeparre et al. [2011] instead of film-to-substrate shear modulus ratio to determine wrinkle frequency. The main idea of Vandeparre et al. is to predict the frequency of cloth wrinkles in the interior based on inward propagation from a nearby compressed, clamped boundary. This scaling law gives good results for simulations involving hanging curtains, or other problems where wrinkling is coming from compression at a boundary (see for instance their Figure 8c). Note that in order to be applicable, the Vandeparre et al. scaling law

requires (1) that the wrinkling in the interior of the cloth is explainable by wrinkles propagating inward from a nearby boundary, and (2) that the wrinkles at the boundary are caused by compression and clamping at that boundary. Wrinkles due to contact and draping (see the dress in Figure 1) in the cloth interior; or shearing of a boundary that is *not* compressed (for instance, see our later experiments: the sheared rectangle, Figure 9, and twisted cylinder, Figure 21); or strain in the interior that vanishes towards the clamped boundary (like in Cerda and Mahadevan’s model problem of a stretched elastic sheet; see Figure 6) violate the modeling assumptions of Vandeparre et al. See our comparison with the extended Zuenko et al.’s approach with Vandeparre’s law on the stretched sheet and sheared rectangle examples in Figure 8 and 10 highlighting these issues.

Finally, there has been work on augmenting fluid simulation with high-frequency wave packets [Jeschke and Wojtan 2017]; these wave packets are “wrinkles” of a sort, but are governed by a very different physics than the shell elasticity considered in our paper.

*Tension Field Theory.* As discussed above in our introduction, *TFT* was successfully applied to the design of inflatable structures [Skouras et al. 2014], and has been used successfully to explain physical and biological phenomena such as the wrinkling of scar tissue [Cerda 2005] and the inflation of parachutes [Baginski 2005]. The convexity of *TFT* was also exploited in order to optimize the design of skin-tight clothing using sensitivity analysis [Montes et al. 2020]. Although not explicitly grounded in tension field theory, several computer graphics techniques for simulating cloth [Choi and Ko 2002; Jin et al. 2017] adopt a similar idea of neglecting or significantly weakening the elastic forces that resist compression. By reducing the likelihood that small strains induce numerically-challenging out-of-plane buckling, these methods are computationally efficient. Likewise weakening the compression forces offers other advantages as well, such as alleviation of membrane locking.

*Data-Driven Approaches.* A final stream of research on efficiently augmenting simulations with fine wrinkles uses data (gathered from the real world; e.g. via motion capture [Löhner et al. 2018], or collected from offline high-resolution simulations) to learn cloth deformation. To accelerate the latter idea, Kim et al. [2013] constructed a compressible secondary cloth motion graph to sample the dynamic space and reduce storage requirements by a factor of 1000. A common idea, especially useful for adding fine details to T-shirts and other relatively skin-tight garments, is to condition the learned deformation on the pose of an underlying mannequin [Hahn et al. 2014; Santesteban et al. 2019; Wang et al. 2010]. This idea can be further applied to transfer of cloth motion from one body shape onto another [Guan et al. 2012]. Different from the pose-based methods, Kavan et al. [2011] and Seiler et al. [2012] learn a dense upsampling operator to obtain more geometric details on a coarse simulated mesh, and does not assume an underlying mannequin.

Although highly efficient, these methods suffer from the usual issues of data-driven strategies: artifacts appear and quality degrades when the method is applied to simulations that require extrapolation rather than interpolation of existing training data. Methods conditioned on mannequin pose thus cannot be applied to free-floating, environmental cloth.

### 3 WRINKLE FIELD MODELING

We model the kinematics of wrinkled thin shells in terms of a coarse *base surface*, augmented by wrinkles parameterized by amplitude and phase fields over the base surface. In this section, we discuss the mathematical details of these ideas. The main result of our analysis is the formulation of the shell energy in Equation (27), written in terms of the base surface and wrinkle-field degrees of freedom; a discretization of this energy will form the basis of our algorithm in Section 4 for computing the static shape of thin shells. Note that for the theory we develop in this section, we do not assume any particular choice of base surface: later in Section 4.1 we will discuss our proposed use of *TFT* simulation to compute the base surface. We then cover discretization and implementation details in Section 5.

Throughout this section we apply tools and ideas from the physics literature on the geometry of wrinkled sheets [Aharoni et al. 2017; Cerda and Mahadevan 2003] and simple curved membranes [Paulsen et al. 2016]. However, to our knowledge, the general analysis of wrinkle shape and energy on curved shells that we perform here, and the resulting formulas in Equation (19) and (27), are novel.

#### 3.1 Preliminaries

We begin with conventions and notation. We assume that we are modeling a hyperelastic shell of fixed thickness  $h$ , and adopt the usual Kirchhoff-Love assumption [Ciarlet 2000] that the shell's 3D volume is formed by extruding a *midsurface*  $\mathbf{r}(u, v) : U \rightarrow \mathbb{R}^3$  a distance  $h/2$  in both directions along the midsurface normal  $\hat{\mathbf{n}}(u, v)$ , where  $U$  is a planar *parameter domain*. For simplicity we assume a St. Venant-Kirchhoff linear constitutive model<sup>1</sup>, in which case it can be shown (see e.g. [Weischedel 2012]) that the elastic energy of the shell, as a function of the midsurface embedding  $\mathbf{r}$ , is given by the *Koiter energy*

$$E = \frac{1}{2} \int_U \left( \frac{h}{4} W_s + \frac{h^3}{12} W_b \right) \sqrt{\det \mathbf{I}_u} \, dudv, \quad (1)$$

$$W_s = \|\mathbf{I}_u^{-1} \mathbf{I} - \mathbf{id}\|_{\text{SV}}^2 \quad (2)$$

$$W_b = \|\mathbf{I}_u^{-1} (\mathbf{II} - \mathbf{II}_u)\|_{\text{SV}}^2 \quad (3)$$

where  $W_s$  and  $W_b$  are stretching and bending energy densities,  $\mathbf{I}$  and  $\mathbf{II}$  are the midsurface first and second fundamental forms, respectively, expressed as  $2 \times 2$  matrices in the parameter coordinates. The tensors  $\mathbf{I}_u$  and  $\mathbf{II}_u$  are “rest” fundamental forms encoding the strain-free shell configuration. In the case where the shell is rest-flat and the domain  $U$  represents the shell's strain-free shape,  $\mathbf{I}_u = \mathbf{id}$  (the identity matrix) and  $\mathbf{II}_u = \mathbf{0}$ . All of our examples are rest-flat, though we will include the rest fundamental forms in the expressions we derive in this section, so that our results extend seamlessly to shells with rest curvature. The norm  $\|\cdot\|_{\text{SV}}$  is a quadratic form depending on the material Lamé parameters  $\alpha$  and  $\beta$ :

$$\|\mathbf{M}\|_{\text{SV}}^2 = \frac{\alpha}{2} \text{tr}^2(\mathbf{M}) + \beta \text{tr}(\mathbf{M}^2). \quad (4)$$

See Appendix A for a more detailed overview of shell theory.

<sup>1</sup>A linear constitutive model is justified for analyzing the energy of wrinkles, since we expect the residual compressive strain in the wrinkled region (after buckling) to be small. We discuss extending our analysis to other hyperelastic materials in Section 7.

Following foundational work in the physics community on wrinkling of sheets [Aharoni et al. 2017], our fundamental modeling assumption is that the shell midsurface can be decomposed into

$$\mathbf{r}(u, v) = \mathbf{r}_b(u, v) + \mathbf{r}_w(u, v), \quad (5)$$

where  $\mathbf{r}_b$  is a low-frequency *base surface*, from which the shell midsurface is constructed by applying a high-frequency *wrinkle correction*  $\mathbf{r}_w$ . This wrinkle correction can have components in both the normal and tangential directions to the base surface. That is, consider a local frame formed by  $\left\{ \frac{\partial \mathbf{r}_b}{\partial u}, \frac{\partial \mathbf{r}_b}{\partial v}, \hat{\mathbf{n}}_b \right\}$  at a point on the base surface. The wrinkled shell  $\mathbf{r}_w(u, v)$  can be expressed as:

$$\begin{aligned} \mathbf{r}_w(u, v) &= f_1(u, v) \frac{\partial \mathbf{r}_b}{\partial u}(u, v) + f_2(u, v) \frac{\partial \mathbf{r}_b}{\partial v}(u, v) + f_3(u, v) \hat{\mathbf{n}}_b(u, v) \\ &= d\mathbf{r}_b(u, v) \mathbf{v}_t(u, v) + f_3(u, v) \hat{\mathbf{n}}_b(u, v) \end{aligned} \quad (6)$$

where  $\mathbf{v}_t = (f_1, f_2)$  encodes the tangential displacement due to wrinkling, and  $f_3$  the normal displacement. Note that  $\mathbf{v}_t$  is a vector field on the *parameter domain*  $U$ , which the embedding Jacobian  $d\mathbf{r}_b$  maps to a tangent vector on the base surface. We will write  $\mathbf{I}_b, \mathbf{II}_b$  for the first and second fundamental forms of the base surface, respectively.

Notice that we are working with *three* distinct surfaces, each with their own geometry: the (2D) parameterization domain, the (3D) base surface, and the (3D) wrinkled midsurface. We will use subscript  $u$  and  $b$  to denote quantities associated with the parameterization domain and base surface respectively.

#### 3.2 Wrinkle Correction from Wrinkle Fields

As in the analyses of Cerda and others [Aharoni et al. 2017; Cerda and Mahadevan 2003; Paulsen et al. 2016], we assume that the wrinkles in the shell have wavelength that can vary spatially over the surface, but that at each point, the wrinkles have a single predominant wavelength and locally coherent wave direction and amplitude. We can thus write the wrinkle correction in terms of the local geometry and deformation of the base surface, together with a non-negative *amplitude* and periodic *phase* field

$$a(u, v) : U \rightarrow \mathbb{R}_{\geq 0}, \quad \phi(u, v) : U \rightarrow S^1 \cong [0, 2\pi).$$

In particular, we can write  $f_3$  explicitly in terms of amplitude and phase:

$$\mathbf{r}_w(u, v) = d\mathbf{r}_b(u, v) \mathbf{v}_t(u, v) + a(u, v) \cos[\phi(u, v)] \hat{\mathbf{n}}_b(u, v) \quad (7)$$

with base surface normal  $\hat{\mathbf{n}}_b$ , where our remaining task is to determine the in-plane part  $\mathbf{v}_t$  of the wrinkle correction (which controls the wrinkle shape profile) as well as the wrinkle amplitude and phase fields  $a$  and  $\phi$ .

A first observation is that, to a good approximation, surfaces wrinkle in order to compensate for surface area lost to compression. If the strain in the wrinkle direction  $\mathbf{w}$  is  $\epsilon_w$ , we should expect the arclength of one wrinkle period to match original material length of that period in the shell at rest:

$$\epsilon_w \approx \frac{a^2}{2} \|\mathbf{d}\phi\|_{\mathbf{I}_u^{-1}}^2. \quad (8)$$

This relationship coupling wave amplitude and frequency has been widely exploited, in both physics and computer graphics [Rohmer



et al. 2010], but it does not tell us the relative magnitude of  $a$  and  $d\phi$ . We observe that this tradeoff is usually *globally* determined by the boundary conditions, strain, and curvature of the base surface. For instance, if you take a rectangular sheet of paper and shear it by displacing its four corners, the sheet will buckle and form a single,  $\Omega$ -shaped bump. But if the entire top and bottom boundaries of the sheet are clamped and the sheet is sheared, a high-frequency pattern of wrinkles appears (see Figure 9). A second issue is that  $d\phi$  must be integrable, i.e., it must be the derivative of some phase function  $\phi : U \rightarrow S^1$  which is well-defined, at minimum, on the patches of  $U$  where  $a > 0$ . It is therefore not possible to arbitrarily prescribe  $d\phi$  using Equation (8), even if the wrinkle amplitude  $a$  could somehow be inferred.

The punchline is that there is no general local rule for choosing  $a$  and  $d\phi$ , without resorting to user guidance or ad-hoc heuristics. We must instead solve for both terms globally over  $U$ .

### 3.3 Fast and Slow Variables

Before dealing with  $a$  and  $\phi$ , we begin by analyzing the in-plane correction unknown  $v_t$ . A tempting idea, which we tried in early unsuccessful experiments, is to solve for  $v_t$  along with  $r_b$ ,  $a$ , and  $\phi$ , as an additional kinematic degree of freedom. To understand why this approach was misguided, consider that even if we assume no in-plane wrinkle correction ( $v_t = 0$ ), the midsurface shape given by Equations (5) and (7) is underdetermined. Wrinkles in the midsurface could be represented either by

- adding wrinkles to the base surface  $r_b$ , and setting  $a = 0$ ;
- using a smooth base surface, but absorbing all midsurface undulation into the amplitude  $a$  of the wrinkle field, while keeping  $\phi = 0$ ;
- using a smooth base surface, and a slowly-varying  $a$ , with the undulations of the wrinkles induced by variations in the phase  $\phi$  over the surface.

Thinking ahead to when we will want to discretize  $r_b$ ,  $a$ , and  $\phi$  on a coarse mesh, it is clear that only the third solution is acceptable, since it is the only one that will not lead to aliasing of the fine wrinkles of  $r$  when  $r_b$ ,  $a$ , and  $\phi$  are restricted to low frequencies.

Carrying this idea further, we classify variables as *slow* or *fast*. Slow variables change at length scales larger than the wavelength of a single wrinkle, whereas fast variables cannot be approximated as constant, even at the wrinkle scale. Fast variables include  $\phi$ ,  $v_t$ , and  $r_b$ . We assume that the following variables are slow:

- the base surface fundamental forms  $I_b$  and  $II_b$ ; equivalently, the base surface strain and curvature;
- wrinkle amplitude  $a$ ;
- wrinkle frequency and orientation  $d\phi$ ;
- the wrinkle shape profile, so that  $v_t$  is a superposition of periodic vector fields with slow direction and amplitude.

Notice that the wrinkle frequency  $d\phi$  is slow despite  $\phi$  itself being fast, and that the characterization of  $a$  as slow and  $\phi$  as fast breaks the symmetry between  $a$  and  $\phi$  both controlling the wrinkle frequency. We will now derive closed-form expressions for  $v_t$ , based on an analysis of shell statics at the scale of a single wrinkle, baking

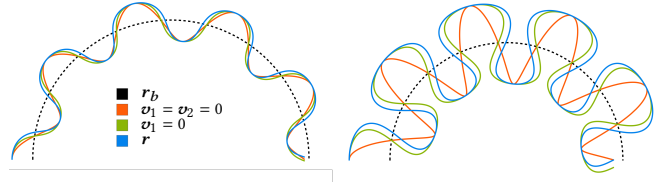


Fig. 3. 2D sketch of the effect that the two in-plane wrinkle correction modes  $v_1$  and  $v_2$  exert on the wrinkle shape, at small (*left*) and large (*right*) amplitudes. The dotted black curve is the base surface. Purely normal wrinkle displacements (orange) strain the surface significantly near where the wrinkles cross the base surface; the  $v_2$  in-plane displacement modifies the wrinkles into a more horseshoe-like shape which equidistributes strain. The  $v_1$  term introduces asymmetry between wrinkles above and below the base surface, allowing the wrinkle shape to adapt to curvature of the base surface.

our assumptions about fast and slow variables into our wrinkle correction model.<sup>2</sup>

### 3.4 In-plane Wrinkle Correction Formulae

Since we assume that the wrinkle shape is a slow variable,  $\|v_t\|$  must be a periodic function of  $\phi$ , at the scale of individual wrinkles. Moreover, we can assume without loss of generality that there is no in-plane translation of the wave crests and valleys:  $\|v_t\|_{\phi=0} = \|v_t\|_{\phi=\pi} = 0$ , since any such translation could be accomplished instead by a phase shift in  $\phi$ . We thus approximate  $v_t$  by the first couple of terms in its discrete sine expansion,

$$v_t \approx v_1 \sin \phi + v_2 \sin 2\phi, \quad (9)$$

with  $v_1$  and  $v_2$  slow vector fields on  $U$ .

There are geometric reasons for expecting both of these terms to be important (see Figure 3). Consider the case of perfectly sinusoidal wrinkles, where  $v_t = 0$ . These wrinkles induce large variations in strain in the shell over one wavelength, with maximum strain at  $\phi = \pi/2, 3\pi/2$  and minimum strain at the wrinkle peaks and valleys. The  $v_2$  term allows a redistribution of material within a wrinkle period to equalize strain.

The  $v_1$  term accounts for strain variation over one wrinkle period, due to curvature of the underlying base surface. Consider for instance a wrinkled cylinder, where the wrinkling direction  $d\phi$  travels azimuthally around the cylinder (similar to the wrinkles shown in Figure 4). Wrinkle peaks are more highly strained than wrinkle valleys, due to the base surface curvature, and the  $v_1$  term allows redistribution of material within a wavelength to compensate. Neglecting  $v_1$  artificially penalizes coarse wrinkles where the base surface is highly curved in the  $d\phi$  direction, and relatively flat in the perpendicular direction (see also the discussion in Section 3.6).

*Finding  $v_1$  and  $v_2$ .* The idea now is to solve for expressions for  $v_1$  and  $v_2$  which minimize the shell's elastic energy density, integrated over a small neighborhood of the surface of size on the order of one

<sup>2</sup>When deriving the elastic energy of the wrinkle field (Equation (19)), we will also assume that the covariant derivative of wrinkle orientation is negligible. This assumption is justified by the observation that, to good approximation, wrinkles align with the direction of principal tension in the shell, and that these directions do not bend significantly within the material plane (since otherwise the material could further deform to relax the tension).

wavelength. Since stretching energy dominates bending energy, it is enough to focus on the stretching energy density (2), which is quadratic in the midsurface strain.

The directions  $\mathbf{v}_1$  and  $\mathbf{v}_2$  are slow variables, and so

$$d\mathbf{v}_t \approx (\mathbf{v}_1 \cos \phi + 2\mathbf{v}_2 \cos 2\phi) d\phi. \quad (10)$$

We can use this expression, as well as the definition  $\mathbf{I} = d\mathbf{r}^T d\mathbf{r}$ , to write down a formula for the strain of the midsurface:

$$\begin{aligned} \mathbf{I}_u^{-1} \mathbf{I} - \mathbf{id} &\approx \mathbf{I}_u^{-1} (\mathbf{I}_b - \mathbf{I}_u + \frac{1}{2} a^2 d\phi^T d\phi) \\ &+ \mathbf{I}_u^{-1} (-2a \mathbf{I}_b + [\mathbf{I}_b \mathbf{v}_1 d\phi]_T) \cos \phi \\ &+ \mathbf{I}_u^{-1} (-\frac{a^2}{2} d\phi^T d\phi + 2[\mathbf{I}_b \mathbf{v}_2 d\phi]_T) \cos 2\phi. \end{aligned} \quad (11)$$

Here we use the notation  $M_T$  to denote the symmetrization  $M + M^T$ . All variables in this expression are slow, except for the trigonometric functions in  $\phi$ . Intuitively, strain, and hence elastic energy, is minimized by minimizing each of the trigonometric coefficients; this idea can be formalized by use of a *coarse-graining operator* [Aharoni et al. 2017]

$$X^{(\text{cg})} := |\Omega|^{-1} \int_{\Omega} X dA \quad (12)$$

which estimates quantities  $X$  averaged over a neighborhood  $\Omega$  of radius comparable to one wrinkle wavelength. Applying coarse-graining to the thin-shell stretching energy density (2) allows us to simplify the energy expression by eliminating oscillatory terms which might be non-zero pointwise, but average to zero (like  $\sin \phi$ ) or a value independent of  $\phi$  (like  $\sin^2 \phi$ ) over a wrinkle wavelength:

$$W_s^{(\text{cg})}(\mathbf{v}_1, \mathbf{v}_2) = \left\| \mathbf{I}_u^{-1} \left( \mathbf{I}_b - \mathbf{I}_u + \frac{1}{2} a^2 d\phi^T d\phi \right) \right\|_{\text{SV}}^2 \quad (13)$$

$$+ \frac{1}{2} \left\| \mathbf{I}_u^{-1} (-2a \mathbf{I}_b + [\mathbf{I}_b \mathbf{v}_1 d\phi]_T) \right\|_{\text{SV}}^2 \quad (14)$$

$$+ \frac{1}{2} \left\| \mathbf{I}_u^{-1} \left( -\frac{a^2}{2} d\phi^T d\phi + 2[\mathbf{I}_b \mathbf{v}_2 d\phi]_T \right) \right\|_{\text{SV}}^2. \quad (15)$$

Notice that  $\mathbf{v}_1$  and  $\mathbf{v}_2$  appear exclusively in terms (14) and (15), respectively, and so we can solve for these variables by minimizing each term independently:

$$\mathbf{v}_1 = a \mathbf{I}_b^{-1} \mathbf{v}, \quad \mathbf{v}_2 = \frac{a^2}{8} \mathbf{I}_b^{-1} d\phi^T \quad (16)$$

where

$$\mathbf{v} = \left( \frac{\alpha}{\alpha + 2\beta} \frac{\text{tr}(\mathbf{I}_u^{-1} \mathbf{I}_b)}{\|\mathbf{w}\|_{\mathbf{I}_u}^2} + \frac{2\beta}{\alpha + 2\beta} \frac{\mathbf{w}^T \mathbf{I}_b \mathbf{w}}{\|\mathbf{w}\|_{\mathbf{I}_u}^4} \right) d\phi^T \quad (17)$$

$$+ 2 \frac{\mathbf{w}^T \mathbf{I}_b \mathbf{w}^\perp}{\|\mathbf{w}\|_{\mathbf{I}_u}^4} (d\phi^\perp)^T; \quad (18)$$

where to de-clutter notation we define  $\mathbf{w} = \mathbf{I}_u^{-1} d\phi^T$  to be the vector field on  $U$  aligned with the wrinkle travel direction, and  $\mathbf{w}^\perp, d\phi^\perp$  the vector field and one-form orthogonal to  $\mathbf{w}$  and  $d\phi$  under the natural  $\mathbf{I}_u$  and  $\mathbf{I}_u^{-1}$  inner products, respectively. The constants  $\alpha$  and  $\beta$  are the Lamé parameters. Notice that on a flat sheet,  $\mathbf{v}_1 = 0$ , confirming the intuition that the role of this term is to modulate the wrinkle shape to accommodate curvature on the underlying base surface. See Appendix C for the derivation of these expressions.

To summarize, we now can replace the wrinkle correction term  $\mathbf{r}_w$  in Equation (5) with an explicit formula in terms of  $\mathbf{r}_b$ ,  $a$ , and  $\phi$ , based on the above derivations of the in-plane part of the correction term:

$$\mathbf{r} \approx \mathbf{r}_b + d\mathbf{r}_b \mathbf{I}_b^{-1} \left( a \sin \phi \mathbf{v} + \frac{a^2}{8} \sin 2\phi d\phi^T \right) + a \cos \phi \hat{\mathbf{n}}_b. \quad (19)$$

### 3.5 Wrinkle Field Energy

Equipped with the formula (19) giving the geometry of the midsurface as a function of the base surface  $\mathbf{r}_b$  and wrinkle field  $a, \phi$ , we can now write down an expression for the elastic energy of the wrinkled shell in terms of these kinematics. Note that, unlike in Section 3.4, where we were operating at the length scale of individual wrinkles, and could assume that slow variables are approximately constant, the elastic energy of the shell is a *global* quantity and we cannot simply eliminate terms depending on e.g. the amplitude field derivative  $da$  which, while they might be negligible locally since  $a$  is slow, might integrate up into a non-negligible energy contribution on the scale of the entire shell.

Deriving the elastic energy of the wrinkle field is, in principle, simply a matter of plugging in Equation (19) into the Koiter energy (1) and simplifying by analyzing the scaling of each energy contribution and eliminating negligible terms. Here we summarize the results of this (in practice, rather involved) procedure (see Appendix D for the derivation).

We consider the stretching and bending terms in the energy separately.

*Stretching Term.* As discussed above, the expression in Equation (11) cannot be used to measure the strain of the midsurface for the purpose of calculating elastic energy, since that formula neglected terms that are non-negligible on the scale of the whole shell. In Appendix D we derive the following energy expression, by coarse-graining and simplifying the first term in Equation (1). We make use of our assumptions in Section 3.3 about fast and slow variables:

$$E_s = \int_U \frac{h}{4} (W_s^1 + W_s^2 + W_s^3 + W_s^4) \sqrt{\det \mathbf{I}_u} du dv \quad (20)$$

$$W_s^1 = \left\| \mathbf{I}_u^{-1} \left( \mathbf{I}_b - \mathbf{I}_u + \frac{1}{2} da^T da + \frac{1}{2} a^2 d\phi^T d\phi \right) \right\|_{\text{SV}}^2 \quad (21)$$

$$W_s^2 = 4a^2 \kappa_\perp^2 \frac{\beta(\alpha + \beta)}{\alpha + 2\beta} \left( \frac{(\mathbf{w}^\perp)^T \mathbf{I}_b \mathbf{w}^\perp}{(\mathbf{w}^\perp)^T \mathbf{I}_u \mathbf{w}^\perp} \right)^2 \quad (22)$$

$$W_s^3 = \frac{a^2}{32} \left\| \mathbf{I}_u^{-1} (d\phi^T da + da^T d\phi) \right\|_{\text{SV}}^2 \quad (23)$$

$$W_s^4 = \frac{1}{8} \left\| \mathbf{I}_u^{-1} da^T da \right\|_{\text{SV}}^2. \quad (24)$$

where the vector field parallel to the wrinkle crests  $\mathbf{w}^\perp$  is defined as in Equation (19), and  $\kappa_\perp = (\mathbf{w}^\perp)^T \mathbf{I}_b \mathbf{w}^\perp / \|\mathbf{w}^\perp\|_{\mathbf{I}_b}^2$  is the normal curvature of the base surface along the  $\mathbf{w}^\perp$  direction.

Although this expression is rather involved, each term can be understood, in retrospect, in the context of the geometry of the fine-scale wrinkling:

- In regions of pure tension, the base surface strain tensor is positive-definite. Since  $da^T da$  and  $a^2 d\phi^T d\phi$  are also positive-semidefinite, *any* amount of wrinkling simply drives up the energy contribution  $W_s^1$ ; this term therefore inhibits wrinkling in regions of pure tension.
- Both  $da$  and  $d\phi$  play an interchangeable role in  $W_s^1$ . However as discussed in Section 3.3, we want wrinkle frequency, and *not* wrinkle amplitude, to absorb compression strain.  $W_s^4$  penalizes large variations in wrinkle amplitude, enforcing that the compression is absorbed by  $d\phi$ , not  $da$  in  $W_s^1$ . In other words,  $W_s^4$  enforces that  $\phi$  is the fast variable, and  $a$  is slow.
- In regions of mixed tension and compression, assuming that amplitude is constant ( $da = 0$ ),  $W_s^1$  is minimized by aligning the wrinkle direction  $d\phi$  with the principal compression direction. Moreover, we therefore recover the intuitive coupling between amplitude, frequency, and strain described in Equation (8).
- The  $W_s^2$  term injects dependence on curvature into the determination of wrinkle amplitude and frequency: high-amplitude wrinkles whose peaks and valleys run along directions of high curvature are penalized. Wrinkles where the isolines of  $\phi$  run along directions of low curvature do not suffer this penalty, even if the surface is highly-curved in the wrinkle travel direction  $d\phi$ . This behavior matches the expected effect of the  $v_2$  on wrinkle shape, as described in Section 3.4.
- Finally,  $W_s^3$  penalizes large changes in amplitude along crests of the wrinkle waves. This matches intuition: amplitude is free to change from one wave to the next, but since wrinkle crests align with directions of tension in the shell, we do not expect variations in wrinkle amplitude *along* one wrinkle.

**Bending Term.** Bending of the wrinkled shell at two scales contributes to the total bending energy: 1) bending of the base surface itself, which contributes energy with formula directly analogous to Equation (3); 2) bending at the fine scale, due to wrinkling. We estimate this latter term from Equation (19) by assuming that the contribution to wrinkle curvature from variations in  $\phi$  dominate any contributions from changes in  $a$  or  $\hat{n}_b$ . In other words, at the fine scale  $\Pi_u \approx 0$  and

$$I_u^{-1} \Pi = I_u^{-1} d^2 r \cdot \hat{n}_b \approx -a \cos \phi I_u^{-1} d\phi^T d\phi. \quad (25)$$

Coarse-graining this second contribution and adding it to the first yields an expression for bending energy,

$$E_b = \frac{h^3}{12} \int_U \left( W_b^1 + W_b^2 \right) \sqrt{\det I_u} \, dudv, \quad (26)$$

$$W_b^1 = \|I_u^{-1} (\Pi_b - \Pi_u)\|_{SV}^2, \quad W_b^2 = \frac{a^2}{2} \|I_u^{-1} d\phi^T d\phi\|_{SV}^2.$$

To sum up, the total elastic energy, in terms of the base surface embedding  $r_b$  and the wrinkle field degrees of freedom  $a$  and  $\phi$ , is

$$E^{\text{wf}} = E_s + E_b. \quad (27)$$

Notice that this energy depends only on the wrinkle frequency and direction ( $d\phi$ ) and not directly on the phase itself ( $\phi$ ); as expected since applying a global phase shift to  $\phi$  changes only fast variables, and should not change the coarse-grained energy of the full shell.

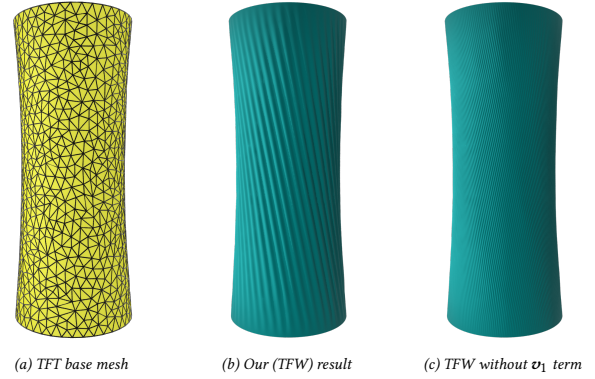


Fig. 4. Simulated cylinder wrinkles using our model (see Section 4) on a cylinder whose boundaries have been clamped and twisted: (a) the 1.3k-vertex base mesh, (b) our result with the  $v_1$  in-plane correction term included, and (c) our result with this term omitted. With in-plane correction, we get 46 waves, which is consistent with results from traditional shell solvers (see Figure 21). Without in-plane correction that accounts for the base surface curvature, our model predicts spuriously high wrinkle frequency (over 180 wrinkles; you may need to zoom in to see them).

We will exploit this invariance when we discretize Equation (27) in Section 5.

### 3.6 Necessity of In-plane Wrinkle Correction

Given the complexity of the stretching energy  $E_s$ , one might look for further simplifications. One source of the complexity are the in-plane wrinkle correction terms in Equation (19). Dropping one or both of these terms (and in doing so, modifying our assumptions about the shape of the wrinkles at the fine scale) significantly modifies the energy  $E_s$ . Without  $v_2$ , there is no longer an asymmetry (from  $W_s^4$ ) between frequency induced by  $d\phi$  and by  $da$ ; in other words, there is no longer enforcement of  $a$  being slow and  $\phi$  being fast. Dropping  $v_1$  significantly changes the curvature term  $W_s^2$ . Both modifications severely degrade the accuracy of wrinkles we recover using the numerical procedure we next describe in Section 4, suggesting that both in-plane correction terms, and all resulting terms in  $E_s$ , are essential to the wrinkle model. In Figure 4 we perform an ablation study on  $v_1$ .

## 4 SOLVING FOR STATIC WRINKLED SHAPE

We now describe an algorithm for using the wrinkle field model of Section 3 to optimize for the static equilibrium shape of draped cloth, inflatable structures, and other wrinkled shells.

One could entertain jointly optimizing the wrinkle field elastic energy in Equation (27) for both the base surface shape as well as the  $a$  and  $\phi$  fields. The challenge with this approach, though, is that our analysis in Section 3 assumed that the base surface was smooth, with curvature a *slow* variable. Optimizing jointly for both the base surface and wrinkle fields without enforcing the slow-ness of  $r_b$  leads to spurious, aliased solutions similar to Figure 2. See Section 7 for more discussion of joint optimization. Here we instead propose a simpler scheme, where we first estimate the base surface shape,



and then fix  $\mathbf{r}_b$  and separately solve for the wrinkle field parameters  $a, \phi$  on that base surface.

#### 4.1 Computing the Base Surface

As argued in the introduction, a low-resolution FEM simulation of the shell is not a good choice for  $\mathbf{r}_b$ : not only does the coarse simulation lack high-frequency wrinkles, it also already possesses *incorrect* aliased, low-frequency wrinkles which our wrinkle field model is not equipped to correct.

For *tension-dominated* problems, optimizing  $\mathbf{r}_b$  using tension field theory is a better choice. A thin shell is tension-dominated if, at most points on the surface, the shell is either in pure tension, or in a mix of tension and compression (it is in these latter regions that wrinkles typically form). Examples of tension-dominated shells include garments hung or draped against gravity, chambers under internal pressure, and fabrics or films tugged by external loads or boundary constraints. Shells under primarily pure compression (such as cloth that is balled up on the floor, or axially-crushed coke cans) are not tension-dominated, and out of scope for our approach.

Tension field theory exploits the fact that wrinkles play little role in force transmission, by formulating a simplified description of local stresses as having only tension components, directed along directions of positive principal stress [Mansfield 1989; Pipkin 1986, 1994; Steigmann 1990]. The theory can be viewed in terms of a modified or *relaxed* variant of the membrane energy density in Equation (2) [Pipkin 1986, 1994]; see Appendix B for an overview. *TFT* optimizes for the coarse envelope of the shell, ignoring local high-frequency wrinkling by allowing the shell to compress without buckling. The *TFT* result is thus ideal as a base surface  $\mathbf{r}_b$  for wrinkle augmentation, using the theory of wrinkle fields developed in Section 3. Since the *TFT* solution expresses only low-frequency geometric features, it remains accurate even for very coarse discretizations of the domain  $U$ ; moreover the modified stretching energy is *convex* and efficient to minimize [Skouras et al. 2014]. Figure 1, left, shows a *TFT* simulation of a dress draped on a mannequin. Notice the smooth, wrinkle-free shape and coarse tessellation. We present many more *TFT* base meshes in Section 6.

**Base Surface Bending Model.** Tension field theory gives a convex replacement for stretching energy, when solving for the base surface shape  $\mathbf{r}_b$ . With an eye towards additional improvements in efficiency and robustness of the base mesh solve, for rest-flat shells we recommend also replacing the elastic bending energy density (Equation (3)) with Bergou et al. [2006]’s Quadratic Bending model,

$$W_b^{\text{qb}} = \alpha \langle \Delta_u \mathbf{r}_b, \Delta_u \mathbf{r}_b \rangle_{I_u}, \quad (28)$$

where  $\Delta_u$  denotes the Laplace-Beltrami operator on  $U$  with respect to the metric  $I_u$ , and  $\alpha$  is the first Lamé parameter. We also adopt Wang et al. [2015]’s correction terms to account for the bending energy at the free shell boundaries.

Advantages of quadratic bending over the full shell bending energy include:

- $W_b^{\text{qb}}$ , being quadratic in the base surface embedding  $\mathbf{r}_b$ , is convex, so that the total elastic energy of the base mesh (*TFT*+Quadratic Bending) is convex;
- the energy is very easy to implement.

Quadratic Bending assumes that the shell is deforming isometrically; this assumption is violated in many of our examples, where the base surface compresses significantly. Nevertheless, we did not observe much change in the shape of the base mesh by substituting Quadratic Bending for Equation (3). For the dress examples in Figures 15 and 17, the Hausdorff distance between the base surface computed using full bending, and Quadratic Bending, is 1% of the dress diameter.

#### 4.2Computing the Wrinkle Fields

Once we have found the base surface, we compute a wrinkle field over that surface by minimizing Equation (27). Here we describe a couple of additional simplifications and reformulations we recommend to simplify this task.

**Approximation of  $\mathbf{w}$ .** Notice that in the  $W_s^2$  term in the wrinkle field stretching energy, the *direction* of  $\mathbf{w}$  is used (both in computing  $\kappa_\perp$  and in the formula for  $W_s^2$  itself), but not its *magnitude*. Moreover, as discussed in Section 3, the main unknown in solving for the wrinkle field is the tradeoff between amplitude and wrinkle frequency. Wrinkle direction is strongly encouraged to align with the principal compression direction of base surface strain, by the energy term  $W_s^1$ . We therefore approximate  $\mathbf{w}$  as a constant (rather than a function of  $d\phi$ ) given by the solution to the generalized eigenvalue problem

$$(I_b - I_u)\mathbf{w} = \lambda I_u \mathbf{w} \quad (29)$$

with most negative eigenvalue  $\lambda$ . In regions where neither eigenvalue is negative (i.e., in regions of pure tension) we simply ignore the  $W_s^2$  term, under the assumption that  $a \approx 0$  in those regions.

**Solving for Frequency Instead of Phase.** In our formulation in Section 3, we assume that  $\phi$  is a periodic function. The need for periodicity is evident even in very simple wrinkling scenarios, such as the drape of a square piece of cloth over a sphere (see Figure 5): rotationally symmetric wrinkles appear around the circumference of the draped cloth, corresponding to a  $\phi$  which continuously linearly increases as you circulate around the draped portion of the cloth (as shown in Figure 5, middle-right). Since Equation (27) depends only on the phase field derivative  $d\phi$  and not on the phase itself, we eschew optimizing for  $\phi$ , and instead borrow from the surface parameterization and stripe pattern optimization [Knöppel et al. 2015] literature the idea of expressing the wrinkle field elastic energy in terms of the one-form  $\omega = d\phi$ . In other words, we solve

$$\begin{aligned} & \arg \min_{a, \omega} E^{\text{wf}}(a, \omega) \\ \text{s.t.} \quad & a \geq 0, \\ & \forall \mathbf{p} \in U, \ a(\mathbf{p}) = 0 \text{ or } \text{curl } \omega(\mathbf{p}) = 0. \end{aligned} \quad (30)$$

Here the first constraint enforces that wrinkles cannot have negative amplitude, and the second ensures that the recovered  $\omega$  can be written, at least locally, as the derivative of a phase field  $\phi$ , everywhere where wrinkles are visible (amplitude is positive). We provide more detail about how to discretize and solve this variational problem in the next section and in Appendix E.

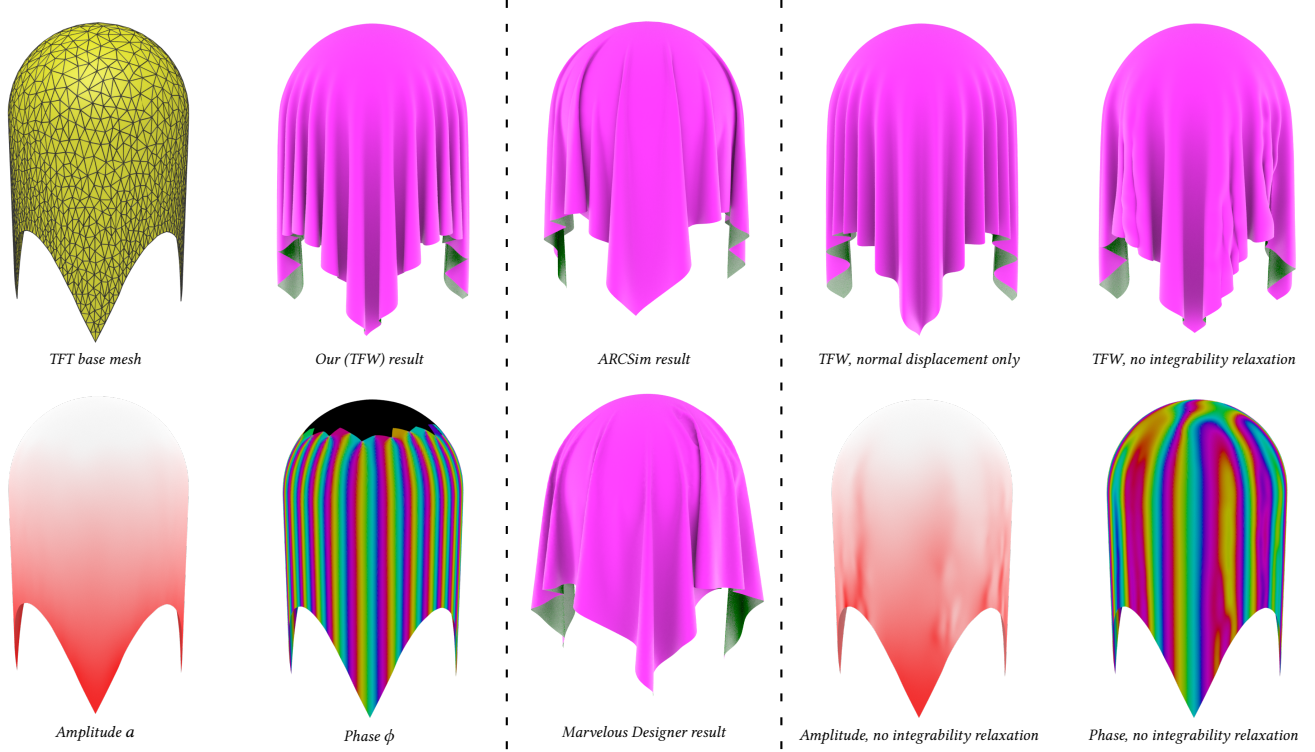


Fig. 5. Draping a square piece of cloth on a sphere. *Left*: the base surface  $\mathbf{r}_b$ , after optimization using *TFT* and Quadratic Bending, on a domain with  $1.6k$  vertices, the surface after we optimize and visualize a wrinkle field on this base surface, and a visualization of the computed amplitude and phase fields  $a, \phi$  over  $\mathbf{r}_b$ . Black regions are where wrinkles don't exist and the phase is undefined. *Middle*: comparison to simulation results from two standard cloth solvers, *ARCSim* and *Marvelous Designer*. To capture all detailed wrinkles, a high resolution simulation mesh was needed (here we use  $72k$  vertices). *Right*: Some ablation experiments on this example. Including the  $\mathbf{v}_2$  in-plane wrinkle correction term when visualizing the wrinkle field is important; here we show the unnatural undulations produced when visualizing the same amplitude and phase fields as in the bottom-left subfigures, but applying only normal displacement ( $\mathbf{v}_2 = 0$ ). We also show the disastrous consequences of failing to relax the integrability constraints on  $\omega$  on pure-tension faces. The recovered amplitude  $a$  and phase  $\phi$  are not the expected smooth, periodic solution since without relaxing integrability near the north pole of the sphere, it is mathematically impossible for an  $\omega$  to circulates around the circumference of the draped portion of the cloth.

## 5 DISCRETIZATION AND SOLVER

Our pipeline for computing and visualizing the static shape of shells using a *TFT* base mesh and a wrinkle field consists of the following steps:

- (1) We triangulate  $U$  into a coarse simulation mesh  $K = (V_u, F, E)$ , with vertices  $V_u = \{\mathbf{v}_u^1, \mathbf{v}_u^2, \dots\}$  (Section E.1).
- (2) We solve for the base mesh embedding  $V_b = \{\mathbf{v}_b^1, \mathbf{v}_b^2, \dots\}$  by minimizing the *TFT* and Quadratic Bending energies (Section 4.1).
- (3) We represent  $a$  as a function on the vertices of  $K$ , and  $\omega$  as a one-form on the mesh edges. We estimate an initial guess for these variables, based on  $V_u$  and  $V_b$  (Section E.2).
- (4) We also locate the faces of  $K$  on which the base mesh exhibits pure tension. We collect these faces into a set of *wrinkle-free faces*  $W$  (Section 5.1).
- (5) We solve the optimization problem in Equation (30) for  $a$  and  $\omega$ , using sequential quadratic programming. We relax the complementarity constraint in this problem by making use of the wrinkle-free faces  $W$  (Section 5.2 and E.3).

- (6) We integrate  $\omega$  to recover the phase field  $\phi$ . Each face maintains a separate value of  $\phi$  for its three vertices (i.e.,  $\phi \in \mathbb{R}^{3|F|}$ ) to support the  $2\pi$ -periodicity of the phase field (Section E.4).
- (7) Finally, we visualize the result by upsampling the base mesh and wrinkle fields using Loop subdivision, and displacing the resulting geometry using Equation (19) (Section E.5).

The output of the above *Tension Field + Wrinkle (TFW)* pipeline is our prediction of the shell static shape, shown in Figure 1, right, and throughout Section 6. In what follows, we discuss the details most critical to understanding and implementing the above pipeline. Further implementation details can be found in Appendix E.

### 5.1 Relaxed Integrability

As we wrote in Equation (30), when minimizing elastic energy with respect to  $a$  and  $\omega$ , we will want to maintain that  $\text{curl } \omega = 0$  everywhere *except* in regions where wrinkles do not exist (amplitude vanishes), so that we can (locally) integrate  $\omega$  into the phase field  $\phi$ . As has been observed many times by researchers in surface parameterization [Bommes et al. 2009; Kälberer et al. 2007], it is absolutely

crucial that singularities, where  $a = 0$  and  $\text{curl } \omega \neq 0$ , be allowed to exist. These singularities are topologically required to recover reasonable one-forms  $\omega$ , even on simple examples. Consider for instance again the example of a square cloth draped on a sphere, shown in Figure 5. The obvious solution  $\omega$  to recover correct wrinkling of the cloth is for  $\omega$  to circulate (like a whirlpool) around the square's center; clearly the path integral of  $\omega$  around the square's boundary here is nonzero. But then it is impossible for  $\text{curl } \omega = 0$  at every point on the square (doing so would violate the fact that every closed one-form on a simply-connected region is exact).

Ideally, our solver would automatically place these singularities in the optimal location (at the north pole of the sphere, in this case), by enforcing the complementarity constraint in Equation (30) on  $a$  and  $\text{curl } \omega$ . However, we are not aware of a simple algorithm for doing so. Instead we make the following observations, leading to a heuristic for placing singularities in reasonable locations a priori:

- in regions under pure tension, the energy term  $W_s^1$  strongly penalizes placing any wrinkles at all. It is reasonable to therefore assume that  $a = 0$  in these regions;
- conversely, a singularity where amplitude vanishes is most likely to be located in a region of pure tension, than in a region of mixed tension and compression or pure compression (where  $W_s^1$  penalizes the lack of wrinkling).

Our heuristic, therefore, is to force  $a = 0$  on a set of *wrinkle-free faces*  $W$  that are under pure tension, and to relax integrability of  $\omega$  on those faces, allowing the solver to anchor singularities at these faces. To compensate for the coarseness of the base mesh (which often only has thousands or hundreds of triangles), we perform some smoothing before classifying triangles as being under pure tension: we average each triangle's most-negative eigenvalue with that of its three neighbors, and if this average is positive, add the triangle to  $W$ . After this procedure, we filter outliers from  $W$  by removing any triangle in  $W$  whose three neighbors are not members, and adding to  $W$  any triangle whose three neighbors are already members.

Figure 5, right, shows the results of *TFW* with and without this relaxation of the curl constraint on wrinkle-free faces. Without relaxation (top-right), we do not recover a reasonable phase field, since the global integrability constraint will force  $\phi$  to zig-zag as you travel around the square's circumference, rather than increasing in a smooth gradient.

## 5.2 Optimization

We discretize  $a$  using piecewise-linear elements over  $K$ , and discretize  $\omega$  as a one-form on the edges  $E$ , in the style of discrete exterior calculus. On triangles where integrability of  $\omega$  is enforced, we do so via the linear constraint  $d\omega = 0$ . We treat first and second fundamental forms  $I_u, II_u$  as constant over each triangle; see Appendix A for the relevant formulas. We can then write the energy of Equation (30) as a sum of integrals over the triangles not in  $W$ ; we compute these integrals using three-point quadrature [Zhang et al. 2009].

After the above discretization, finding  $a$  and  $\omega$  amounts to minimizing a degree-eight polynomial objective function, subject to linear equality constraints (enforcing integrability) and inequality constraints (enforcing positive amplitudes). We solve this problem

via sequential quadratic programming, applying the NASOQ QP solver [Cheshmi et al. 2020] to compute the constrained descent directions at each iteration. For more details about the discretization and solver implementation, please see Appendix E.3.

## 6 EVALUATION AND DISCUSSION

Below we consider the behavior of the *TFW* pipeline on a wide range of test examples designed to investigate both fidelity to experimental results as well as comparison to results obtained by large degree-of-freedom, traditional shell simulators. For the latter we utilize three representative simulators:

- As a best-in-class academic library for shell modeling we apply *ARCSim* [Narain et al. 2013, 2012], a widely deployed dynamics cloth simulator. Critically for our comparison, *ARCSim* offers the option of adaptive remeshing, which allows traditional shell simulation to capture fine wrinkle details with lower degree-of-freedom meshes by only refining specific triangles. To apply *ARCSim* for solving statics we apply critically damped time-steps to equilibria. In the following, for each *ARCSim* example, we will indicate when the adaptive remeshing option is applied or not.
- As a high-performance commercial cloth simulator we also include comparisons with *Marvelous Designer* [2020], a widely-used industrial garment-design tool that deploys its own proprietary statics physics solver for predicting garment drape. In the following we will denote this simulator as *MD*.
- Finally, to compare with a baseline, consistent shell model we implement a thin shell statics simulator that solves for the St. Venant-Kirchhoff material model with constant-strain stretching elements and mid-edge bending (Morley) elements [Chen et al. 2018; Grinspun et al. 2003; Weischedel 2012]. In the following we will denote this simulator as *StVK*. We made a best effort to optimize this baseline code, while restricting computation to the CPU; 70% of each Newton iteration is consumed by our chosen third-party linear solver (SuiteSparse's CHOLMOD [2008]) which gives us confidence that our code is free of gross inefficiencies. See Appendix F.1 for a detailed timing breakdown.

The full data set of meshes used in our evaluation is provided as supplemental material and we will release the source code of both our *TFW* and *StVK* reference implementations.

### 6.1 Draping Behavior

We begin by analyzing the qualitative behavior of *TFW* on a series of real-world draping examples exhibiting complex wrinkling geometries and nontrivial wrinkle topology. Notice that unlike data-driven wrinkle-recovery methods, our method is purely model driven. No user guidance nor extra data beyond material parameters and boundary conditions are required. For examples with this level of complexity, exact comparisons do not make sense (many of these drapes likely have multiple metastable states, even before taking differences in how each solver models physics and frictional contact into account). In later sections we will perform exact comparisons on simpler model problems where experimental and/or analytic results are known; here, instead, we show that *TFW* results are comparable



qualitatively to those generated by traditional solvers. To set the material parameters, we first choose one of the predefined fabrics given in *ARCSim* (Navy Sparkle Sweat), then derive approximate physical parameters from the provided bending and stretching stiffnesses. We get the following material parameters for *TFW* and *StVK*: density  $200\text{kg/m}^3$ , 0.3 Poisson's ratio, 0.1MPa Young's modulus and thickness 1mm. *MD* does not expose these same material parameters, so we instead manually select a material from *MD*'s predefined palette of material types that gives us closest-matching results to *StVK/ARCSim*.

*Sphere Drape.* We first drape a square cloth ( $1\text{m}\times 1\text{m}$ ) over a sphere of radius 0.2m. In Figure 5 we demonstrate the resulting base surface  $\mathbf{r}_b$ , from the joint *TFT* and Quadratic Bending modeling on a domain with 1.6k vertices, and the resulting *TFW*-generated surface after we visualize the wrinkle field on the base surface. We also show results for *ARCSim* (with adaptive remeshing) and *MD*: for the former, the final adaptive mesh has 32k vertices. For *MD*, we manually search for the coarsest mesh able to resolve the cloth's wrinkles; in this case, 72k vertices. The resulting *TFW* wrinkles are qualitatively similar in frequency and amplitude, especially compared to *MD*, although, as discussed, it is hard to infer ground truth here since we see broken symmetry in both *ARCSim* and *MD* results, where two sides of the front corner have differing shapes, suggesting that there are many metastable solutions. One noticeable difference is that *TFW* does not "puff out" as much as the other two simulations; and the wrinkles do not collapse and flatten under their own weight in the manner that is seen in the *ARCSim* and *MD* results. See Section 7 for more discussion of these limitations.

*Garments.* We next consider garment drapes. We pose two dress patterns (from the Berkeley Garment Library) and a pair of pants on mannequins. Figures 15, 16, and 17 detail results and relevant mesh resolutions for all solvers. These examples all demonstrate complex geometry and topology; notice that our results recover wrinkle fields with widely-varying frequency and direction, and that despite the coarse base mesh employed, *TFW* generates results qualitatively close to the three traditional, high-degree-of-freedom cloth solvers. The last column in Figures 15, 16, and 17 present final results when employing *ARCSim* with *adaptive remeshing* enabled. Here these adapted-mesh examples give a sense as to why traditional simulations generally succeed in recovering wrinkle details only when the simulation mesh is at quite high resolution (over 40k vertices were needed for these three garment examples). Our *TFW* model generates comparable wrinkle patterns with just a few thousands vertices per example (3.4k, 1.4k and 4k for the two dresses and pants respectively).

## 6.2 Resolution Analysis

We consider the sensitivity of traditional shell simulators to their mesh resolution, and contrast with the *TFW* pipeline's low-resolution requirements for its base mesh. To do so we study the asymmetric dress as well as a new example, a twisted cylinder (see Figure 21): here a cylindrical (but rest-flat) shell of radius 1m, height 5m, and thickness 0.1mm, clamped at the top and bottom boundaries, is

twisted at the top boundary by  $10^\circ$  while keeping the distance between the top and bottom boundaries constant. This example uses 0.44 Poisson's ratio and 10MPa Young's modulus. The cylinder example is ideally suited for a convergence analysis, as the number of wrinkles around the cylinder is objective, discrete, and easily-counted. We can thus use the number of wrinkles as a proxy for how rapidly each simulation method converges under refinement.

We first test *TFW* on irregular meshes for these examples with increasing resolutions. For the dress, we start from a base mesh with only 382 vertices and monotonically increase the mesh resolution. We observe that our model converges to a consistent shape at  $\approx 1.4\text{k}$  vertices (see Figure 20). Applying the same test on the twisted cylinder yields an even more impressive result, where a consistent shape emerges at  $\approx 180$  vertices (see Figure 21).

We next probe the behavior of the traditional solvers by performing the following experiments: (1) we run *StVK*, for irregular meshes of increasing resolution; (2) we do the same for *ARCSim*, with adaptive remeshing disabled in order to force use of a mesh with given resolution; and (3) we run *ARCSim* with adaptive remeshing enabled. See Figures 19 and 21 for results. Clear aliasing of high frequencies are evident, as expected, at coarse resolutions. For the twisted cylinder, it is difficult to say how close *StVK* and *ARCSim* are to converging, as the wrinkle number keeps changing for them even for meshes with over 96k vertices. Similarly, adaptive remeshing also has trouble determining a final, consistent state. Here we show the result of adaptive meshing after waiting two days (8,000 simulation steps). This example illustrates that adaptive meshing is not a silver bullet for effectively resolving wrinkle features: adaptivity does not provide appreciably smaller meshes in examples like this when fine wrinkles cover a large portion of the shell surface. Moreover, this example illustrates how local refinement can introduce artifacts in the final shape: earlier decisions about where to refine biases how the shell later deforms; see the unequal wrinkle spacing in Figure 21, left column.

Unlike the cylinder example, a careful convergence analysis is not possible for the dress example, as we observe that for all methods, the same code and the same mesh with different initializations can converge to different solutions. We can determine that the traditional simulators all appear to roughly converge for this example with meshes in the range of about 20–40k vertices for each solver. We also observe that *ARCSim*, both with and without remeshing, yields qualitatively different solutions for the dress example. Despite the lack of quantitative certainty, we can confidently conclude, however, that the resolution required to reproduce wrinkles with frequency and fidelity qualitatively equal to those predicted by our method generally requires approximately an order of magnitude higher resolution than *TFW* for all three simulators (*ARCSim*, *StVK*, and *MD*) for both the cylinder and dress examples.

## 6.3 Meshing Independence

We next probe mesh-dependence of *TFW* by testing four differently generated simulation meshes for the same dress-drape example. We consider meshes generated by: (a) direct Delaunay triangulation (via the Triangle library [Shewchuk 1996]); (b) upsampling from a lower-resolution Delaunay mesh, where we use a modified Loop

subdivision to keep vertices along seams of the garment stitch pattern unchanged; (c) downsampling from a finer Delaunay mesh, where we also keep the seam vertices unchanged; and (d) extracting a mesh from *ARCSim* after allowing it to take one step with its adaptive remeshing enabled. We run our *TFW* pipeline for each of the four simulation meshes. Figure 18 demonstrates that, despite these variations in triangulation, the final wrinkled shapes are consistent.

However, as the tension field theory provides no penalty on compression, a given amount of compression applied to region of a surface may not equidistribute within that region, so that compressive strain may end up concentrated in only a narrow strip of triangles. In practice we do observe some inconsistency ( $\approx 15\%$ ) in our model (see the third and fourth columns of Figure 21) for regular, structured meshes, due to this phenomenon. For example, while for the irregular Delaunay mesh (1.3k vertices) we get 46 waves, for the regular mesh generated by diagonalizing a regular  $N \times K$  quad mesh, where  $N$  refers to the number of uniform azimuthal samples, and  $K$  the number of uniform axial samples, we get varying wave numbers: for the ones diagonalized to align with the twist direction, we have 39 and 38 waves, and for the ones with opposite diagonalization, this number is 40, where we tested with  $N = 80$ ,  $K = 12$ , and  $N = 120$  and  $K = 9$ .

In turn, we observe that traditional solvers can also suffer from artifacts on regular meshes aligned unfavorably with the wrinkles. First consider the *StVK* simulator. The last three columns in Figure 21 illustrate this effect, where we test the *StVK* simulator with mesh resolutions ranging from 6k vertices and then doubling resolution until 96k. The seventh column in Figure 21 shows the results for irregular Delaunay meshes, where the wrinkle frequency increases from 10 to 19. For the regular mesh diagonalized along the twist direction, the number of wrinkles oscillates (the eighth column in Figure 21), whereas for the meshes diagonalized in the opposite way, this number increases from 12 to 20 (the ninth column in Figure 21). For such problems, we observe that even *ARCSim* has convergence issues, ultimately producing an irregular wrinkle pattern with 69k vertices; see the first two columns in Figure 21.

#### 6.4 Accuracy

To further investigate accuracy of the *TFW* model we now consider examples where the wrinkling behavior is known either through analysis or experiment.

**Stretched Sheets.** In a pioneering experiment, Cerda and Mahadevan [2003] studied wrinkling in a thin sheet whose left and right boundaries are clamped and then pulled apart. The sheet compresses in the perpendicular (vertical) direction due to Poisson’s ratio and horizontal wrinkles appear. They provide scaling laws for determining the wrinkle frequency and amplitude. A range of successive works have extended the detailed study of this problem analytically and numerically with varying elasticity models [Healey et al. 2013; Li and Healey 2016; Wang et al. 2018].

We simulate this model problem with a rectangular sheet of size  $0.25\text{m} \times 0.1\text{m}$ , with thickness  $0.1\text{mm}$ , Poisson ratio  $\nu = 0.5$  and Young’s modulus  $Y = 1\text{MPa}$ . *TFW* produces wrinkles with 5 wave periods and a peak amplitude  $0.42\text{mm}$  on a base mesh of 522 vertices. These values are within 20% of the results determined by Wang et

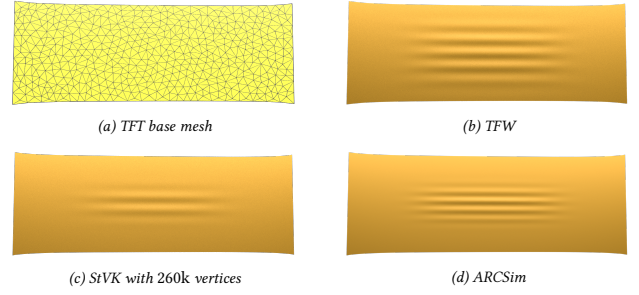


Fig. 6. Simulation of wrinkles in a highly stretched sheet [Wang et al. 2018]. (a) *TFT* base mesh with 522 vertices. (b) *TFW* predicts correct wrinkling to within 20% of theoretical values with the given base mesh. (c) *StVK* will also predict qualitatively-correct wrinkles, however only starting with meshes at a resolution of 260k-vertices and higher. (d) *ARCSim*’s result, with adaptive remeshing. The final mesh has 13k vertices.

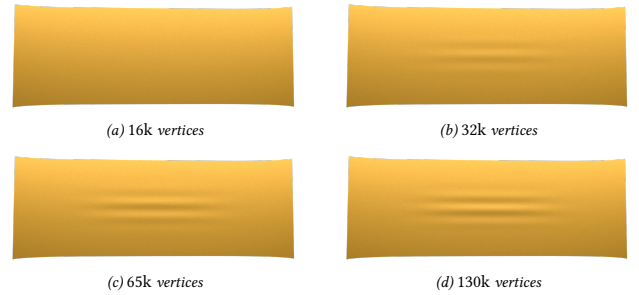


Fig. 7. Simulation of wrinkles in a highly stretched sheet [Wang et al. 2018] using the *StVK* model on a sequence of Delaunay meshes. (a) No waves appear in the mesh with 16k vertices. (b) 3 waves with  $0.17\text{mm}$  peak amplitude appear when mesh resolution increases to 32k. (c) A 65k-vertex mesh yields 3 waves but  $0.27\text{mm}$  peak amplitude. (d) Mesh with 130k vertices ends up with 3 waves and  $0.31\text{mm}$  peak amplitude.

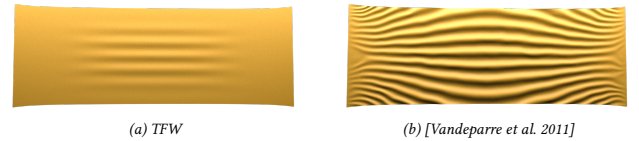


Fig. 8. Comparison of (a) our method and (b) wrinkles generated using the scaling law of Vandeparre et al. [2011] for the model problem of a highly stretched sheet [Wang et al. 2018]. In examples where the wrinkles are not caused by wrinkles propagating inward from a clamped boundary, the analysis of Vandeparre et al. does not apply.

al. [2018] ( $0.35\text{mm}$  for peak amplitude). In comparison, *ARCSim* with adaptive remeshing enabled, generates a mesh of 13k vertices,

producing 5 wrinkles and peak amplitude of 0.34mm. See Figure 6 for visualization of these results.

Despite the seeming simplicity of this example set-up, traditional simulation methods struggle to correctly predict the wrinkling behavior here and so it provides a simple “unit-test” for accuracy. *StVK* can produce wrinkles, but success is sensitive to resolution. We test *StVK* on a sequence of Delaunay meshes of the rectangle, doubling the number of vertices each time. Wrinkles first appear at 32k vertices, but the predicted pattern still does not converge to a consistent result even as we extend *StVK* to a 260k vertex mesh. For this highest-resolution mesh *StVK* generated a solution with 3 wave periods and peak amplitude 0.33mm (Figure 7 and Figure 6(c)). Given that the amplitude of the wrinkles is quite small compared to the dimension of the overall structure, membrane locking [Chapelle and Bathe 2011] is likely responsible for artificially stiffening the material, especially on “coarse” meshes.

We also use this model problem to probe the scaling law proposed by Vandeparre et al. [2011], and suggested by Zuenko et al. [2019] for use in cases where a wrinkling shell is not bonded to a volumetric substrate. As shown in Figure 8, although the Vandeparre et al. scaling law is suitable for predicting frequency cascades in wrinkles that propagate inward from clamped boundaries, it is not suitable for predicting more general wrinkling patterns, even in simple cases like the stretched sheet experiment.

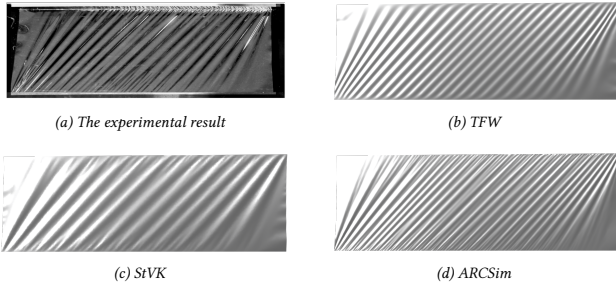


Fig. 9. A thin rectangular sheet sheared in the horizontal direction. (a) The experiment result from [Wong and Pellegrino 2006] yields 19 wrinkles, (b) our method (*TFW*) produces 25 wrinkles, (c) *StVK* generates 13 wrinkles, and (d) *ARCSim* gives 26 waves.

**Sheared Rectangles.** Wong and Pellegrino [2006] study the wrinkle profile of a sheared rectangle whose top and bottom boundaries are clamped and sheared in the horizontal direction while its left and right boundaries are left free. We reproduce their experimental set-up in simulation, where *TFW* generates 25 wrinkles with 1k vertices. As in the previous experiments, we search for the minimum-resolution of the simulation for which *ARCSim* and *StVK* does not exhibit significantly degraded wrinkling; under this methodology *ARCSim* gives 26 wrinkles using an 18k-vertex mesh, and *StVK* produces 13 wrinkles on a 20k-vertex mesh. Wong and Pellegrino report 19 wrinkles in the actual experiment. Visually, as presented in Figure 9, *TFW* shows consistency with the real world example, where both the simulated wavelength and wrinkle direction are

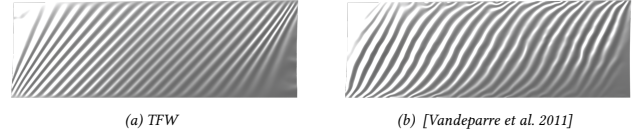


Fig. 10. Comparison of wrinkles generated using (a) our method and (b) the scaling law of Vandeparre et al. [2011] for shearing a rectangular sheet [Wong and Pellegrino 2006]. Again, in this example, the interior wrinkles are not caused by boundary wrinkles that propagate inward from a compressed, clamped boundary (here the boundary is sheared inextensibly), so the analysis of Vandeparre et al. does not apply. You can see unnatural wrinkles near the top and bottom boundaries in (b).

well-aligned with the experimental result. Finally, notice from the experimental photograph that our assumption of a single dominant wavelength is valid over most of the wrinkled surface, and that our model recovers these dominant wrinkles; however there is also a thin boundary layer near the clamped edges of the sheet where a superimposed second frequency of wrinkles can be seen. Our model cannot currently resolve these secondary wrinkles; see Section 7 for discussion of how our model might be extended to the case of multiple superimposed wrinkles.

For this problem, we also compute the wrinkles using the scaling law proposed by Vandeparre et al. [2011], which is suggested by Zuenko et al. [2019] as an extension of their method for shells with no substrate. Similar to the stretched sheet experiment, we observe unnatural behavior near the boundaries (see Figure 10).

**Inflated Structures.** Inflatable structures exhibit a wide range of complex wrinkling behaviors that have been modeled with direct application of tension-field simulations [Skouras et al. 2014]. Here we consider *TFW*’s behavior on a range of inflated examples by augmenting our *TFT* model with Skouras et al.’s [2014] pressure force. In each of the following three examples we inflate a balloon domain along their boundaries: an annulus, disk, and rectangle.

Sewing two annuli (Poisson’s ratio  $\nu = 0.44$ , Young’s modulus  $Y = 10\text{MPa}$ ) together (inner radius 0.04m and outer radius 0.1m) yields a torus. We observe fine wrinkling around the outer equator of the torus when simulating *TFW* with a 2k vertex base mesh. We perform the same experiment with *StVK*, observing that wrinkles are not fully resolved until we reach a mesh of 40k vertices (Figure 11).

Two disks (radius 0.1m) sewn along their boundary yields the classic Mylar balloon, with wrinkles around the equator. Here we use the same material parameters as in the previous example. Interestingly, for this example, inspecting a real-world balloon reveals features at two scales: a small number of coarse *creases* appear equally spaced around the equator, with fine *wrinkles* in between. Both features can also be seen in the *StVK* simulation of the Mylar balloon, and in previous simulations of this problem using adaptive subdivision finite elements [Vetter et al. 2014]. Here we find that *StVK* requires a simulation mesh of at least 40k vertices to resolve these features. *TFW* also produces a result with both scales of features, despite its wrinkling model assuming only a single wrinkle



frequency (Figure 11). This behavior is surprising at first until we observe that the creases already begin in the *TFT* base mesh. The *TFW* wrinkle model then appropriately augments these creases with the fine wrinkles. We are currently uncertain when and why these sharp creases appear in the base mesh, but do verify that this is not an artifact due to the choice of our bending model: these sharp creases appear even if there is zero bending energy (Figure 12). We do observe that resolving the sharp creases requires a relatively high base mesh resolution relative to correctly predicting the compression field and fine wrinkling; in this case we get sharp creases if our base mesh has at least 5k vertices. There has been relatively little work in the physics literature studying *TFT* in this regime; better understanding of creasing behavior in *TFT* base meshes and likewise its corresponding implications for *TFW* wrinkling remains promising future work.

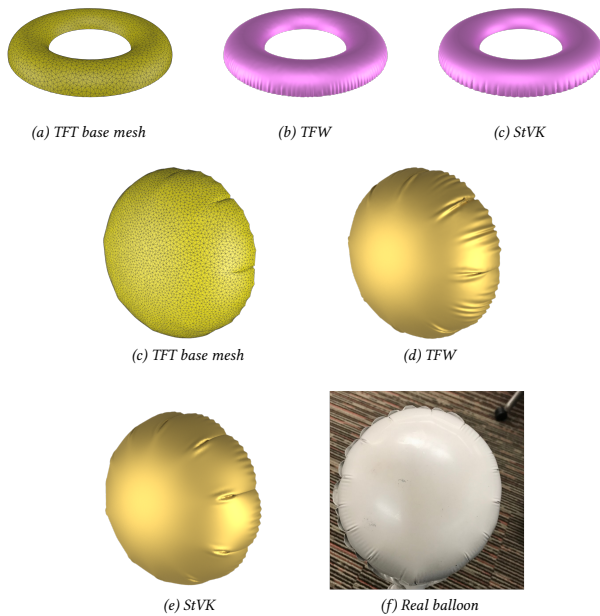


Fig. 11. The simulated results of inflated annulus and disk experiments. *Top*: from left to right, a 2k base mesh, the *TFW* solution, and the *StVK* solution, on a mesh with 40k vertices. *Middle left*: the simulated *TFT* base mesh for the disk balloon, with 5k vertices. *Middle right*: the wrinkled mesh produced by *TFW*. *Bottom left*: simulated result on 40k-vertex mesh. *Bottom right*: the real-world balloon.

Two rectangular patches (width 1.4m, height 0.74m), with thickness 1mm, Poisson’s ratio 0.3 and Young’s modulus 0.1MPa, when inflated yield the classic “teabag” shape. Understanding the coarse shape and wrinkling of this teabag is a classical problem in mathematics [Pak and Schlenker 2010; Paulsen 1994]. *TFW* with a 1k-vertex mesh predicts fine wrinkles around the boundary of the teabag, but here we do not additionally recover the coarse creases evident in a 30k *StVK* simulation (Figure 13). Again, as in the “Mylar balloon” discussion above, this behavior is closely related to how well the base mesh resolves creasing. Here we only start to see creases in the base *TFT* mesh at finer resolutions ( $\approx 30k$  vertices).

Lastly, we simulate wrinkles on the Teddy bear example from Skouras et al. [2014]. This bear has more complex inflated geometry—it is sewn from 17 planar patches. *TFW* generates wrinkles in a 2k-vertex base mesh that are qualitatively similar to those that appear in a *StVK* simulation starting at around 98k vertices (Figure 14).

## 6.5 Performance and Numerical Convergence

In this work we have focused on accurately capturing the wrinkling features of thin shell elastica with as few computational degrees of freedom as possible. In so doing we have derived the *TFW* model which successfully obtains fine wrinkling behavior over a wide range of examples, with generally an order-of-magnitude less degrees-of-freedom than standard shell methods; our method generally requires an order of magnitude less computation time as well. We summarize the resolution and timing of our experiments in Table 1. We ran our experiments on a desktop with a 8-core Intel Core i9-9900K CPU, clocked at 3.6 GHz and 128 GB of memory.

*Resolution Comparisons.* We compare *TFW*, *StVK*, and *ARCSim* in terms of the lowest-resolution mesh required in order to resolve the wrinkles without significant degradation. For *TFW* and *StVK*, we begin with a lowest-resolution Delaunay mesh, and solve for the static wrinkled shape; we double the resolution of the mesh and repeat the experiment until the wrinkle pattern converges to a consistent shape, or until the simulation takes more than a week to terminate (in which case we halt the experiment). The resolution in the table is the lowest at which the result is the consistent wrinkled shape. We indicate with a “−” superscript the simulations where the coarsest-resolution simulation is already consistent (so that potentially even lower-resolution consistent simulations are possible) and with a “+” superscript those simulations that became too computationally expensive before yielding consistent results.

For *ARCSim*, we enable adaptive remeshing and list the final mesh resolution in Table 1. *ARCSim* requires specification of additional parameters, including most notably a minimum triangle size, which significantly affects the quality of the solved static shape. We set frame time to 0.04s, frame steps to 8, and end time to 40s; for the minimum triangle size, we try 0.01m, then 0.005m and finally 0.001m, accepting the first result in which the wrinkles are not aliased. Note that we do not report *ARCSim* results for the inflated structures, since *ARCSim* does not implement a pressure force. For the other examples, we disable the *ARCSim* “popfilter” module, as well as the “collision” module for examples without collisions.

*Timing Comparisons.* Computation of the *TFW* base mesh is now reduced via our choice of *TFT* and quadratic bending to an entirely convex problem on a coarse mesh. Similarly, computation of the wrinkle field is then likewise a small, sparse optimization of our discretized wrinkle energy subject to sparse curl-constraints and simple bound constraints on the amplitude degrees of freedom. In our experiment, the reduction in mesh resolution directly translates into reduction in computational cost, when comparing *TFW* to traditional solvers. That said, apples-to-apples comparisons are not straightforward: unlike for the resolution experiments above, timing comparisons depend significantly on the low-level implementation details of both our method and the baselines. Moreover, it is not

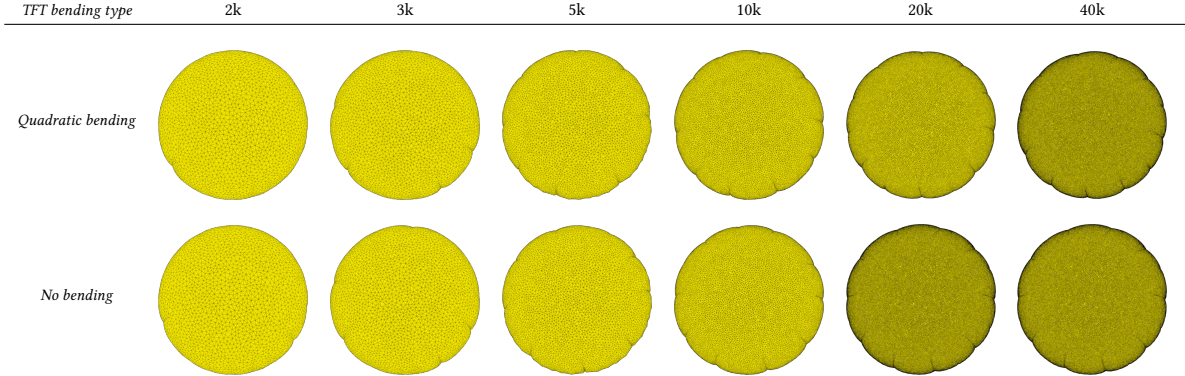


Fig. 12. The simulated tension field result of the inflated disk experiment. From left to right the resolutions are 2k, 3k, 5k, 10k, 20k, 40k. *Top row*: simulated *TFT* model with quadratic bending. Numbers of sharp creases are 1, 4, 8, 10, 11, 12. *Bottom row*: the simulated *TFT* model without bending term; the corresponding wrinkle numbers are 2, 5, 8, 10, 10, 10.

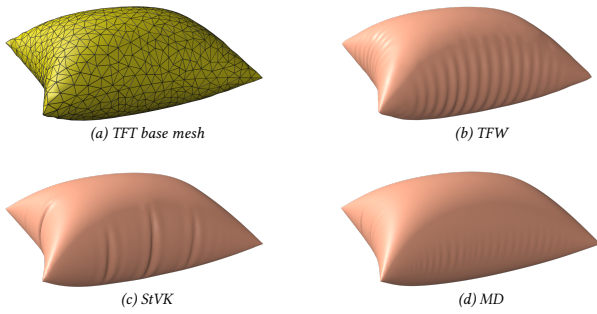


Fig. 13. The simulated result of inflated teabags: (a) the *TFT* base mesh with 936 vertices; (b) shape recovered using *TFW*; (c) *StVK* simulation on a 30k-vertex mesh, and (d) *Marvelous Designer* simulation with 30k vertices. Among all these results, *StVK* achieves the most natural shape (sharp creases + small wrinkles). *TFW* will need a much higher resolution to capture these sharp creases in the *TFT* base mesh.

clear how to determine a termination condition that is consistent across all methods: gradients with respect to vertex positions, and with respect to wrinkle field variables  $a$  and  $\omega$ , have different units and cannot be compared.

Despite the above difficulties, we provide some timing data in Table 1. For each experiment that does not involve collision response<sup>3</sup>, we run *TFW* and *StVK* for 1000 SQP iterations, or until the residual infinity norm is below  $10^{-6}$ , whichever comes first. For both methods, 1000 iterations is far more than necessary to reach a visually-stable static shape. We visually inspect the intermediate configurations and select the earliest iteration where the solution

<sup>3</sup>We do not perform timing comparisons for the examples involving contact, since our (very naive) contact solver is very slow and dominates the cost of both the *TFW* and *StVK* optimizations.

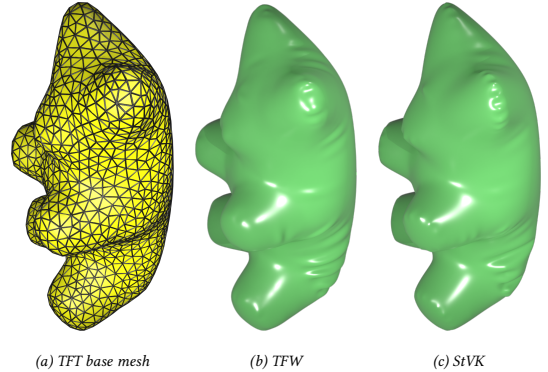


Fig. 14. Inflated Teddy bear: (a) the 2k-vertex *TFT* base mesh, (b) corresponding *TFW* result, and (c) simulated result with *StVK* (98k vertices). We can see *TFW* predicts a similar wrinkle patterns as *StVK*, but requires 50 times fewer vertices.

matches the visually-stable shape; the wall-clock time of this iteration is listed as the *stable time* in Table 1. We use the *TFT* base mesh as the initial guess for both *TFW* and *StVK* (and the timings in the table do not include this preprocessing step); the time needed to compute the initial guess is negligible compared to the subsequent solve times. See Appendix F.1 for timing numbers.

Although inherently subjective, this methodology allows us to give some sense of the relative performance of *TFW* versus baselines, and we observe that the manually-selected visually-stable frame matches a “dogleg” in each example’s stationarity residual plots. See Appendices F.2 and F.3 for additional data and residual plots for all examples listed in the table. We observe that *TFW* offers a speedup ranging from 1.28x (for the torus) to 345.1x (for the

stretched sheet experiment) compared to *StVK*. For all the experiments, *TFW* succeeds in reaching a visually-stable wrinkled shape within one minute.

Since commercial shell solvers like ABAQUS emphasize accuracy and generality over performance, and since we are not aware of any established computer graphics software for shell statics, we used our own implementation of Morley shell elements (based on the implementation hints provided by Grinspun et al. [2003] and Weischedel [2012]; see Appendix A) as the *StVK* baseline. We made best-effort optimizations to improve performance of both the *TFW* and *StVK* algorithms; in both cases the majority of the time spent each iteration is in the third-party solver (NASOQ [Cheshmi et al. 2020]) in the case of *TFW*, and SuiteSparse’s parallel implementation of supernodal sparse Cholesky decomposition [Chen et al. 2008] in the case of *StVK*. See Appendix F.1 for more information including data about the timing breakdown within each optimization iteration.

Finally, in Figure 22, we visualize the progress of the *TFW* and *StVK* solvers on a wall-clock time axis, to give an equal-effort comparison of the two methods. On the timeline (which is in log scale) we indicate the range of times during which each simulation has not yet reached a visually-stable state in red. The visually-stable time (as listed in Table 1) corresponds to the transition on the timeline from red to blue. In cases where we terminate a simulation early, due to having reached very low stationarity residual ( $10^{-6}$ ), the termination time corresponds to the transition from blue to green. We show representative stills of each simulation; each still was sampled at the wall clock time of its left edge. Please see Appendix F.3 for raw data and discussion of the blue-to-green transition, and the supplemental material for videos showing  $a, \omega$  (for *TFW*) and vertex displacement (for *StVK*) versus wall clock time.

## 7 CONCLUSION, LIMITATIONS, AND FUTURE WORK

We have introduced *TFW*, a new model and algorithm for high-fidelity modeling of wrinkling thin shells suitable for low-resolution computational meshes. The key insight is that by decoupling the wrinkled shape into slow and fast variables, and deriving an approximate elastic energy in terms of the slow amplitude and frequency variables, high-resolution, physically-principled detail can be added to a wrinkle-free coarse mesh without aliasing. We have demonstrated *TFW*’s ability to generate realistic and often predictive results across a wide range of challenging examples with an order-of-magnitude less degree of freedoms and speedups of between 1.28x and 345.1x compared to traditional finite elements.

Although our solver implementation in Section 5 serves as proof of concept for applying wrinkle fields as a new approach to static simulation of fine wrinkling, many avenues of future work remain before a wrinkle-field approach is industry-ready as a practical replacement for the current, triangle-element-based approach:

**Performance.** This paper largely focused on foundational theory, rather than low-level performance optimization (nevertheless, in the experiments of Section 6.5 we were able to achieve significantly faster performance with *TFW* compared to baseline *StVK*, due to the vastly coarser mesh needed by *TFW*). To compete with commercial GPU-based cloth solvers, *TFW* would need additional performance improvements. The performance bottleneck for *TFW*

(see Appendix F.1 for a breakdown) is the SQP solve for amplitude and frequency (Section 5.2); at a minimum, a high-performance implementation of *TFW* would need to use a parallelized QP solver on the GPU instead of the CPU-based NASOQ library. Other potential optimizations include starting from a better initial guess for amplitude and phase (based on additional analysis of the physics of wrinkling, or provided by a data-driven approach), simplifying the discretization of the reduced-order elastic energy (for example, by replacing three-point quadrature of the integrals in Equation (30) with simpler expression derived using Discrete Exterior Calculus), or replacing our SQP strategy for minimizing elastic energy with a different optimization strategy.

**Two-way Base Surface-Wrinkle Field Coupling.** We derived an approximated elastic energy  $E^{\text{wf}}$  (Equation (27)) that involves both a base surface  $\mathbf{r}_b$  and wrinkle field variables  $a, \omega$ . In this paper, we solve first for the base surface using tension field theory (*TFT*), and then fix this surface and optimize independently for the wrinkle variables. A natural question is whether the *TFT* base mesh is truly optimal in terms of minimizing  $E^{\text{wf}}$ . As we can see from Fig. 5, in some cases it is clear that the *TFT* solution does not “puff out” enough: the wrinkling of the draped cloth corrugated the surface, which in turn penalizes bending of the cloth perpendicular to the wrinkles, near where the cloth breaks contact with the sphere, so that the draped cloth has a more conical than cylindrical coarse shape. As discussed earlier, jointly optimizing  $E^{\text{wf}}$  for both  $\mathbf{r}_b$  and  $a, \omega$  does not work, since  $E^{\text{wf}}$  assumes that the base surface strain and curvature are slow variables, which is no longer necessarily true if  $\mathbf{r}_b$  is allowed to vary arbitrarily. One idea might be to parameterize  $\mathbf{r}_b$  using differential coordinates, a low-resolution subdivision surface, or some other space of deformations that ensures the base surface strain and curvatures stay slow. Another potential approach to two-way coupling would be to alternate solving for  $\mathbf{r}_b$  and the wrinkle field, where an “effective” rest curvature is computed for  $\mathbf{r}_b$  based on the current wrinkle field at each iteration.

**Other Base Surface Limitations.** Even when using *TFT* to compute the base surface, sometimes the base surface mesh can have large variations in strain, or defects such as inverted or collapsed triangles, which can cause *TFW* to struggle. For example, the relatively slow performance of *TFW* on the inflated torus example is due to high noise in the amount of compression in the corresponding base mesh. The root cause of this noise is that *TFT* has multiple possible solutions in regions of compression. For example, if a square piece of cloth is compressed in the horizontal direction, both the deformation where the entire cloth has equal compression strain (desirable), and the deformation where only a thin vertical column of the cloth compresses while the rest of the cloth translates isometrically (undesirable), are minimizers of the *TFT* energy. The quality of the *TFT* base surface might be improved by adding some regularization terms to the *TFT* energy, or by incorporating two-way coupling of the base surface with the wrinkle patterns, as discussed above.

**Phase Ambiguity.** Our approach computes wrinkle amplitude and frequency, and then solves for phase  $\phi$  as a post-process, with  $d\phi = \omega$ . This recovery procedure can determine phase only up to a global phase shift,  $\phi \rightarrow \phi + k$ ; this shift cannot be determined from the

Models	<i>StVK</i>			<i>ARCSim</i>	<i>TFW</i>		
	#verts	#iterations	stable time (s)	#verts	#verts	#iterations	stable time (s)
sphere drape	—	—	—	32k	1.6k <sup>−</sup>	30	11.08
symmetric dress	40k	—	—	8k	1.4k	20	8.18
asymmetric dress	52k <sup>−</sup>	—	—	13k	3.4k	30	23.19
pants	60k <sup>−</sup>	—	—	15.8k	4k	25	38.94
stretched sheet	260k <sup>+</sup>	13	886.80	11k	522 <sup>−</sup>	13	2.57
sheared rectangle	20k <sup>+</sup>	220	673.11	13k	1k <sup>−</sup>	50	9.63
torus	40k	7	57.29	—	2k	55	44.81
balloon	40k	10	76.52	—	5k	30	52.33
teabag	30k	24	148.37	—	936 <sup>−</sup>	13	1.98
teddy	98k	10	251.91	—	2k <sup>−</sup>	75	26.29
twisted cylinder	96k <sup>+</sup>	120	1528.68	68k	688	400	50.94

Table 1. Timing and resolution information for the examples we show in the paper. Reported mesh resolutions are generally the coarsest-possible that do not exhibit degradation of the wrinkle shape; see main text for details of the methodology. A superscript <sup>−</sup> means that the lowest-resolution mesh we tried produced good results (so that the true minimum-required resolution might be lower); a superscript <sup>+</sup> indicates that the highest-resolution mesh we tried (listed in the table) failed to produce acceptable results after one week of simulation; we did not continue to probe higher resolutions for these examples. We did not simulate the inflatable structures using *ARCSim* as the simulator does not include a pressure model. For examples that do not involve contact, we also report timing information for *TFW* and *StVK*. We list the iteration number and wall-clock time for when each simulation converges to a visually-stable result (see main text for methodology).

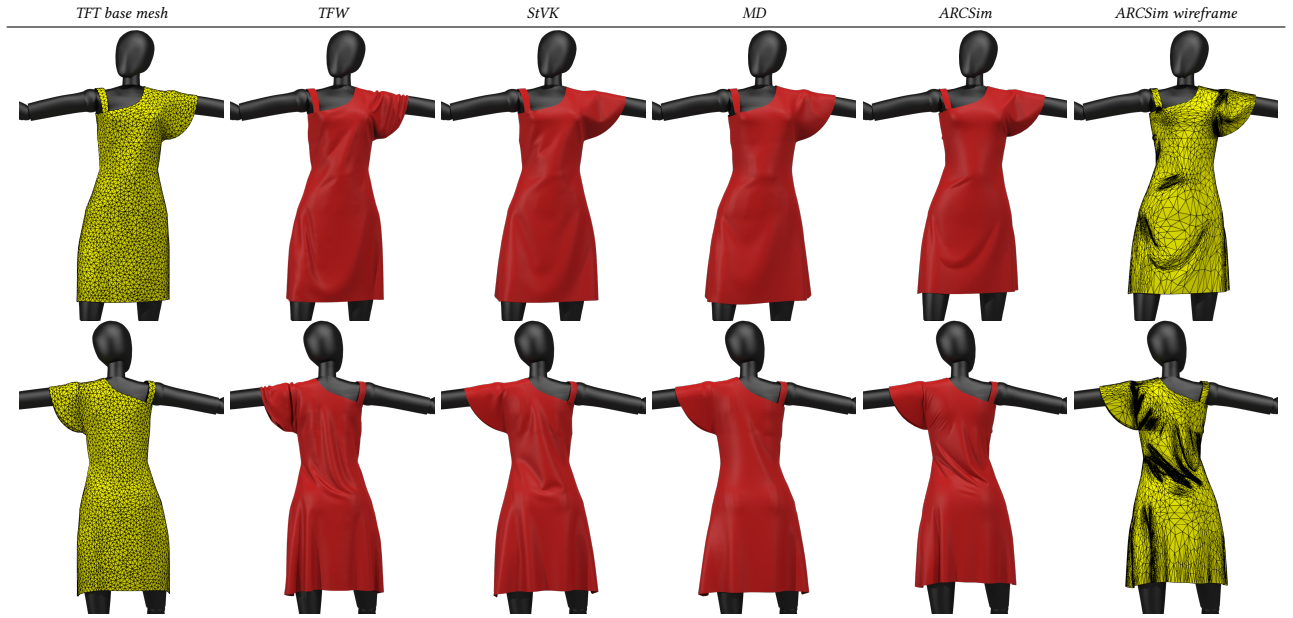


Fig. 15. An asymmetric dress draped over a mannequin. The first row shows the front view of the dress after simulation and the second row shows the back view of the same dress. From left to right: the 3.4k-vertex base mesh, simulated using *TFW*; the simulated wrinkled shell using *TFW* on that base mesh; the result from *StVK* on a Delaunay mesh with 50k vertices; the result from *MD* on a 50k-vertex mesh; the result from *ARCSim*, with adaptive remeshing enabled; a wireframe rendering of that same *ARCSim* result (the mesh has 13k vertices). For *StVK* and *MD*, we tuned mesh resolution to be about as coarse as possible without causing significant degradation in the wrinkle pattern.

wrinkle field. For statics problems the shift is unimportant, as it affects the precise wrinkled geometry, but not coarse-scale features of the wrinkled surface such as wrinkle orientation, amplitude, and frequency. The phase shift ambiguity does mean that the wrinkled surface  $\mathbf{r}_w$  cannot be used to measure convergence of the wrinkle field optimizations, in experiments such as we did in Section 6.5. Any

use of wrinkle fields for dynamics would also need to account for this phase ambiguity across time, as otherwise, sudden changes in the global phase shift would be perceived as “popping” in animations.

*Dynamics.* This paper considers only shell statics. Extending *TFW* to dynamics is a natural follow-up direction; the simplest approach





Fig. 16. A mannequin wearing simulated pants. The first row shows the front view of the pants after simulation and the second row shows the back view of the same pants. From left to right: a 4k-vertex base mesh, simulated using *TFT*; the simulated wrinkled shell using *TFW* on that base mesh; the result from *StVK* on a Delaunay mesh with 60k vertices; the result from *MD* on a 60k-vertex mesh; the result from *ARCSim*, with adaptive remeshing enabled; a wireframe rendering of that same *ARCSim* result (the mesh has 15k vertices). For *StVK* and *MD*, we tuned mesh resolution to be about as coarse as possible without causing significant degradation in the wrinkle pattern.

would be to animate the base mesh and then add wrinkles quasi-statically. The main new feature needed for such quasi-static simulation is a method for solving for a temporally-coherent global phase shift in  $\phi$  at each time step (see previous point) so that wrinkles appear to change smoothly over time. More interesting, and more challenging, would be to equip the wrinkles themselves with inertia, and implement two-way coupling of the wrinkles and base mesh (see above).

**Collisions.** We currently consider only collisions of the *TFT* base mesh against the environment. There are many interesting directions for future work improving collision handling of wrinkled surfaces, such as taking into account collisions when solving for the wrinkle field (so that wrinkles in contact with other objects have flattened shape), detecting and resolving self-contact of the wrinkled surface (where  $\mathbf{r}_w$  self-collides but  $\mathbf{r}_b$  does not), anisotropic friction models that account for the wrinkling direction and amplitude when

a wrinkled surface slides against another object, etc. Most of these future directions will first require research into handling two-way coupling of the base surface and the wrinkle field (see above).

**More General Constitutive Models.** Real-world textiles are woven or knitted, and obey macroscopic constitutive laws that are far more complex than the *StVK* isotropic material we assume in our derivations. Theoretically, there is no obstruction to extending our derivation in Section 3 to other constitutive laws. For orthotropic or anisotropic linear materials, the St. Venant-Kirchhoff material norm  $\|\cdot\|_{SV}$  would need to be replaced by a different quadratic norm, new expressions for the in-plane correction terms (Equation (16); see also Appendix C and Equation (36)) would need to be derived for the new norm, and the in-plane energy term (Equation (22)) updated accordingly. In our derivation we exploited symmetries of the *StVK* material norm to simplify these calculations, and for other



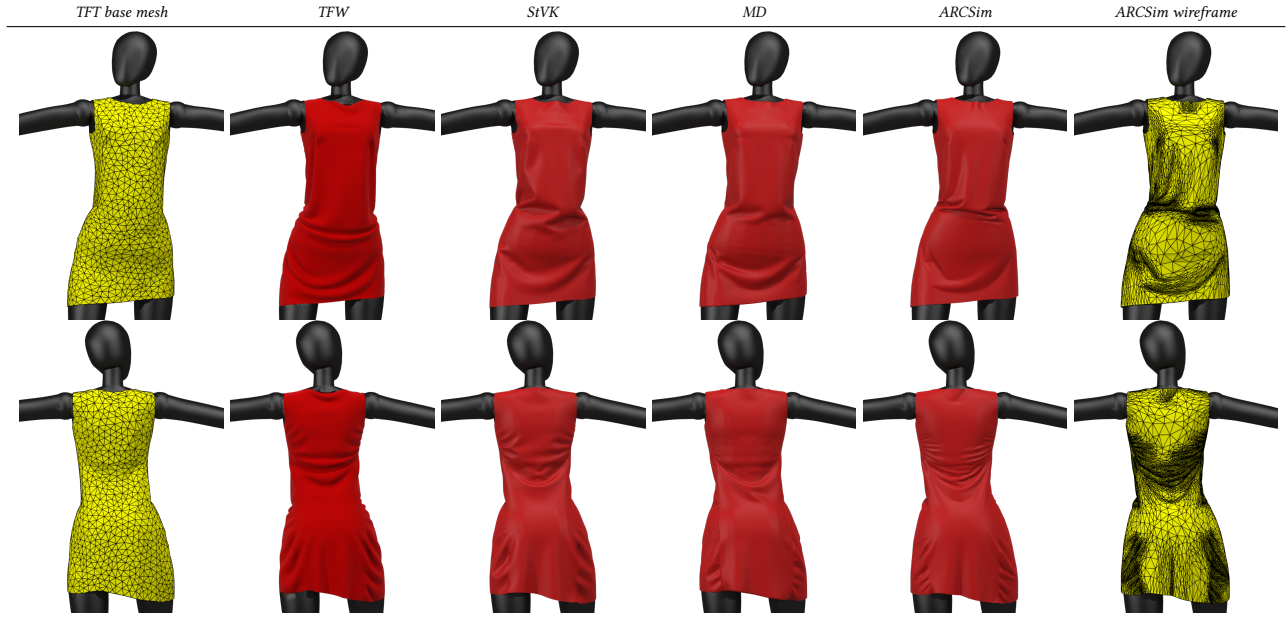


Fig. 17. A symmetric dress draped over a mannequin. The first row shows the front view of the dress after simulation and the second row shows the back view of the same dress. From left to right: a 1.4k-vertex base mesh, simulated using *TFT*; the simulated wrinkled shell using *TFW* on that base mesh; the result from *StVK* on a Delaunay mesh with 40k vertices; the result from *MD* on a 40k-vertex mesh; the result from *ARCSim*, with adaptive remeshing enabled; a wireframe rendering of that same *ARCSim* result (the mesh has 7.8k vertices). For *StVK* and *MD*, we tuned mesh resolution to be about as coarse as possible without causing significant degradation in the wrinkle pattern.

constitutive models, the corresponding equations may be less pleasant. Nonlinear materials would require more extensive rederivation, while still following the roadmap we sketch in Section 3. However, we suspect general nonlinear constitutive models would produce results that differ little from those of their linearizations, since *all* hyperelastic material models reduce to an (isotropic or anisotropic) linear material in the small-strain limit, and note that in regions of wrinkling, the strain in the compression direction is typically small since most of it has been relieved by buckling.

*More General Wrinkling Model.* Several of our modeling decisions in Section 3 might be revisited and extended in future work: for instance, we assume a single predominant wrinkling frequency at each point on the shell, and while this assumption appears to hold in physical experiments, we do know that near boundary layers, real-world shells often exhibit a second, finer frequency of wrinkling (see the sheared rectangle experiment in Figure 9 for instance). One could also explore more general expressions for the wrinkling waveform, for instance by adding additional frequencies to  $f_t$ , or changing the normal displacement from a cosine wave to other shapes such as sinc waves [Evgeny and Harders 2019].

*Injecting Noise and Asymmetry.* In some examples, such as the sphere drape (Figure 5) and the inflated structures, the *TFW* wrinkles can look “too symmetric” compared to wrinkles in real materials, where imperfections and defects break symmetry. Real wrinkles often also collapse under gravity and fold on themselves (noticeable for the sphere drape, in particular), phenomena we do not attempt to

model in this paper. In future work, post-processing could be done on the wrinkle field to emulate these effects, while maintaining the correct wrinkle frequency and amplitude.

## ACKNOWLEDGMENTS

We thank Rahul Narain for guidance on setting up *ARCSim* to run our comparison experiments; and Chris Wojtan, David Levin, Eleni Katifori, Bernhard Thomaszewski and Josh Vekhter for valuable discussions about wrinkling and the geometry processing of vector fields. This project was inspired by discussions at the Bellairs Research Institute’s Workshop on Computer Animation. We thank the NSF (IIS-1910274) and Adobe Inc. for their generous support of this research.

## REFERENCES

- Hillel Aharoni, Desislava Todorova, Octavio Albarran, Lucas Goehring, Randall Kamien, and Eleni Katifori. 2017. The smectic order of wrinkles. *Nature Communications* 8 (07 2017), 15809. <https://doi.org/10.1038/ncomms15809>
- Frank Baginski. 2005. On the Design and Analysis of Inflated Membranes: Natural and Pumpkin Shaped Balloons. *SIAM J. Appl. Math.* 65, 3 (2005), 838–857. <https://doi.org/10.1137/S0036139903438478>
- David Baraff and Andrew Witkin. 1998. Large steps in cloth simulation. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. ACM, New York, NY, USA, 43–54. <https://doi.org/10.1145/280814.280821>
- Miklós Bergou, Saurabh Mathur, Max Wardetzky, and Eitan Grinspun. 2007. TRACKS: Toward Directable Thin Shells. *ACM Trans. Graph.* 26, 3 (July 2007), 50–es. <https://doi.org/10.1145/1276377.1276439>
- Miklós Bergou, Max Wardetzky, David Harmon, Denis Zorin, and Eitan Grinspun. 2006. A Quadratic Bending Model for Inextensible Surfaces. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing (SGP '06)*. 227–230. <https://doi.org/10.1145/1281957.1281987>

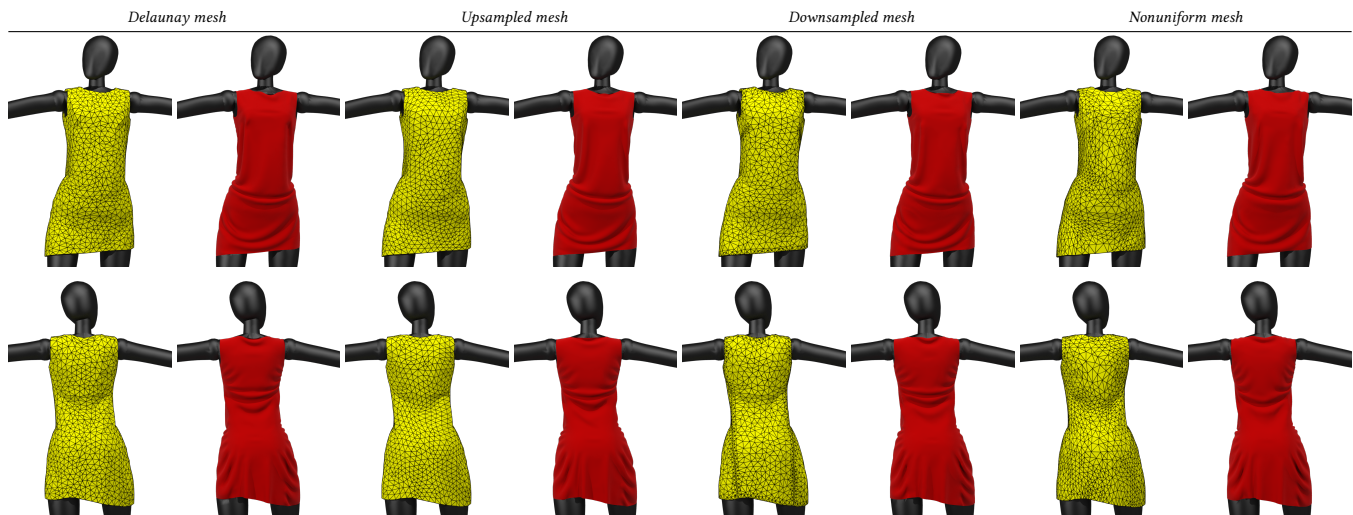


Fig. 18. A symmetric dress draping over a mannequin in terms of different triangulation, but similar mesh resolution ( $1.7k \pm 0.3k$  vertices). The first row shows the front view of the dress after simulation and the second row shows the back view of the the same dress. From left to right: *TFT* simulated result of a 1406-vertex Delaunay mesh. The corresponding wrinkled shape using our wrinkled-field model on that base mesh; *TFT* simulated result of a mesh with 1399 vertices. This mesh is generated by Loop upsampling a lower resolution Delaunay mesh. The corresponding wrinkled shape using our wrinkled-field model on that base mesh; *TFT* simulated results of a mesh with 1959 vertices, downsampled from a higher resolution Delaunay mesh (keeping the stitching boundary unchanged). The corresponding wrinkled shape using our wrinkled-field model on that base mesh; *TFT* results of the mesh extracted after one-step arcsim remeshing process with 1981 vertices. The corresponding wrinkled shape using our wrinkled-field model on that base mesh. Our wrinkled-field model yields a consistent final result with respected to these different triangulation.

- David Bommers, Henrik Zimmer, and Leif Kobbelt. 2009. Mixed-Integer Quadrangulation. *ACM Trans. Graph.* 28 (07 2009). <https://doi.org/10.1145/1576246.1531383>
- Eddy Boxerman and Uri Ascher. 2004. Decomposing Cloth. (2004), 153–161. <https://doi.org/10.1145/1028523.1028543>
- Juan J. Casafranca, Gabriel Cirio, Alejandro Rodríguez, Eder Miguel, and Miguel A. Otaduy. 2020. Mixing Yarns and Triangles in Cloth Simulation. *Computer Graphics Forum* 39, 2 (2020), 101–110. <https://doi.org/10.1111/cgf.13915>
- Enrique A. Cerda. 2005. Mechanics of scars. *Journal of Biomechanics* 38, 8 (2005), 1598–1603. <https://doi.org/10.1016/j.jbiomech.2004.07.026>
- Enrique A. Cerda and Lakshminarayanan Mahadevan. 2003. Geometry and Physics of Wrinkling. *Physical review letters* 90 (03 2003), 074302. <https://doi.org/10.1103/PhysRevLett.90.074302>
- Dominique Chapelle and K.J. Bathe. 2011. *The Finite Element Analysis of Shells - Fundamentals - Second Edition*. Springer. 410 pages. <https://doi.org/10.1007/978-3-642-16408-8>
- Hsiao-Yu Chen, Arnav Sastry, Wim M. van Rees, and Etienne Vouga. 2018. Physical Simulation of Environmentally Induced Thin Shell Deformation. *ACM Trans. Graph.* 37, 4, Article 146 (July 2018), 13 pages. <https://doi.org/10.1145/3197517.3201395>
- Yanqing Chen, Timothy A. Davis, William W. Hager, and Sivasankaran Rajamanickam. 2008. Algorithm 887: CHOLMOD, Supernodal Sparse Cholesky Factorization and Update/Downdate. *ACM Trans. Math. Softw.* 35, 3, Article 22 (Oct. 2008), 14 pages. <https://doi.org/10.1145/1391989.1391995>
- Kazem Cheshmi, Danny M. Kaufman, Shoaib Kamil, and Maryam Mehri Dehnavi. 2020. NASOQ: Numerically Accurate Sparsity-Oriented QP Solver. *ACM Trans. Graph.* 39, 4, Article 96 (July 2020), 17 pages. <https://doi.org/10.1145/3386569.3392486>
- Kwang-Jin Choi and Hyeon-Seok Ko. 2002. Stable but Responsive Cloth. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques* (San Antonio, Texas) (SIGGRAPH '02). 604–611. <https://doi.org/10.1145/566570.566624>
- Philippe G. Ciarlet. 2000. *Theory of Shells, Volume 3 (Mathematical Elasticity)*. North Holland.
- Marvelous Designer CLO Virtual Fashion Inc. 2020. MARVELOUS DESIGNER, Best Realistic Cloth Making Program For 3D Artists. <https://www.marvelousdesigner.com>
- Zuenko Evgeny and Matthias Harders. 2019. Wrinkles, Folds, Creases, Buckles: Small-Scale Surface Deformations as Periodic Functions on 3D Meshes. *IEEE Transactions on Visualization and Computer Graphics* PP (05 2019), 1–1. <https://doi.org/10.1109/TVCG.2019.2914676>
- Russell Gillette, Craig Peters, Nicholas Vining, Essex Edwards, and Alla Sheffer. 2015. Real-Time Dynamic Wrinkling of Coarse Animated Cloth. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (Los Angeles, California) (SCA '15). 17–26. <https://doi.org/10.1145/2786784.2786789>
- Eitan Grinspun, Anil N. Hirani, Mathieu Desbrun, and Peter Schröder. 2003. Discrete Shells. In *Symposium on Computer Animation*, D. Breen and M. Lin (Eds.). The Eurographics Association. <https://doi.org/10.2312/SCA03/062-067>
- Eitan Grinspun, Petr Krysl, and Peter Schröder. 2002. CHARMS: A Simple Framework for Adaptive Simulation. *ACM Trans. Graph.* 21, 3 (July 2002), 281–290. <https://doi.org/10.1145/566654.566578>
- Peng Guan, Loretta Reiss, David Hirshberg, Alexander Weiss, and Michael Black. 2012. Drape: Dressing any person. *ACM Transactions on Graphics - TOG* 31 (07 2012). <https://doi.org/10.1145/2185520.2185531>
- Fabian Hahn, Bernhard Thomaszewski, Stelian Coros, Robert W. Sumner, Forrester Cole, Mark Meyer, Tony DeRose, and Markus Gross. 2014. Subspace Clothing Simulation Using Adaptive Bases. *ACM Trans. Graph.* 33, 4, Article 105 (July 2014), 9 pages. <https://doi.org/10.1145/2601097.2601160>
- Timothy J. Healey, Qingdu Li, and Ron-Bin Cheng. 2013. Wrinkling Behavior of Highly Stretched Rectangular Elastic Films via Parametric Global Bifurcation. *Journal of Nonlinear Science* 23 (2013), 777–805. <https://doi.org/10.1007/s00332-013-9168-3>
- Stefan Jeschke and Chris Wojtan. 2017. Water Wave Packets. *ACM Trans. Graph.* 36, 4, Article 103 (July 2017), 12 pages. <https://doi.org/10.1145/3072959.3073678>
- Ning Jin, Wenlong Lu, Zhenglin Geng, and Ronald P. Fedkiw. 2017. Inequality Cloth. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (SCA '17). Article 16, 10 pages. <https://doi.org/10.1145/3099564.3099568>
- Felix Kälberer, Matthias Nieser, and Konrad Polthier. 2007. QuadCover - Surface Parameterization using Branched Coverings. *Comput. Graph. Forum* 26 (09 2007), 375–384. <https://doi.org/10.1111/j.1467-8659.2007.01060.x>
- Jonathan M. Kaldor, Doug L. James, and Steve Marschner. 2008. Simulating Knitted Cloth at the Yarn Level. In *ACM SIGGRAPH 2008 Papers* (Los Angeles, California) (SIGGRAPH '08). Article 65, 9 pages. <https://doi.org/10.1145/1399504.1360664>
- Ladislav Kavan, Dan Gerszewski, Adam Bargteil, and Peter-Pike Sloan. 2011. Physics-Inspired Upsampling for Cloth Simulation in Games. *ACM Trans. Graph.* 30 (07 2011), 93. <https://doi.org/10.1145/2010324.1964988>
- Doyub Kim, Woojong Koh, Rahul Narain, Kayvon Fatahalian, Adrien Treuille, and James F. O'Brien. 2013. Near-Exhaustive Precomputation of Secondary Cloth Effects. *ACM Trans. Graph.* 32, 4, Article 87 (July 2013), 8 pages. <https://doi.org/10.1145/2461912.2462020>

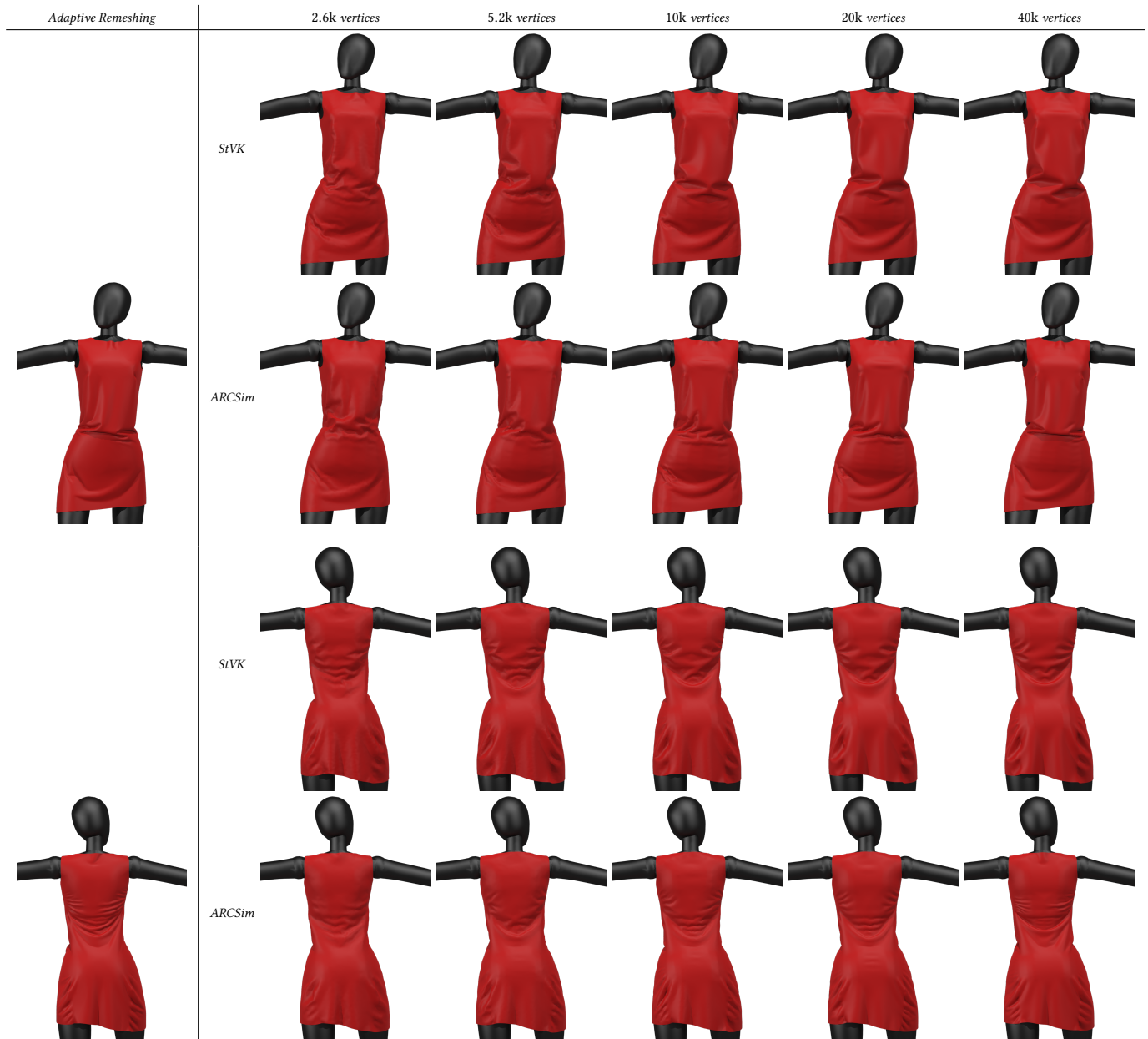


Fig. 19. We simulate the draping of a dress with different mesh resolutions, using *StVK* and *ARCSim* (with (*left column*) and without (*second and fourth row*) remeshing). The heading of each column indicates the mesh resolution of that column. The simulations appear to converge at about 20–40k vertices for each solver.

Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2015. Stripe Patterns on Surfaces. *ACM Trans. Graph.* 34, 4, Article 39 (July 2015), 11 pages. <https://doi.org/10.1145/2767000>

Zorah Löhner, Daniel Cremers, and Tony Tung. 2018. DeepWrinkles: Accurate and Realistic Clothing Modeling. In *Computer Vision – ECCV 2018*. Springer International Publishing, Cham, 698–715. [https://doi.org/10.1007/978-3-030-01225-0\\_41](https://doi.org/10.1007/978-3-030-01225-0_41)

Jie Li, Gilles Daviet, Rahul Narain, Florence Bertails-Descoubes, Matthew Overby, George Brown, and Laurence Boissieux. 2018. An Implicit Frictional Contact Solver for Adaptive Cloth Simulation. *ACM Transactions on Graphics* 37, 4 (Aug. 2018), 1–15. <https://doi.org/10.1145/3197517.3201308>

Qingdu Li and Timothy J. Healey. 2016. Stability boundaries for wrinkling in highly stretched elastic sheets. *Journal of the Mechanics and Physics of Solids* 97 (2016), 260 – 274. <https://doi.org/10.1016/j.jmps.2015.12.001> SI-Pierre Suquet Symposium.

Gurobi Optimization LLC. 2020. Gurobi Optimizer Reference Manual. <http://www.gurobi.com>

Eric Harold Mansfield. 1989. *The Bending and Stretching of Plates* (2 ed.). Cambridge University Press. <https://doi.org/10.1017/CBO9780511525193>

Juan Montes, Bernhard Thomaszewski, Sudhir Mudur, and Tiberiu Popa. 2020. Computational Design of Skintight Clothing. *ACM Trans. Graph.* 39, 4, Article 105 (July 2020), 12 pages. <https://doi.org/10.1145/3386569.3392477>



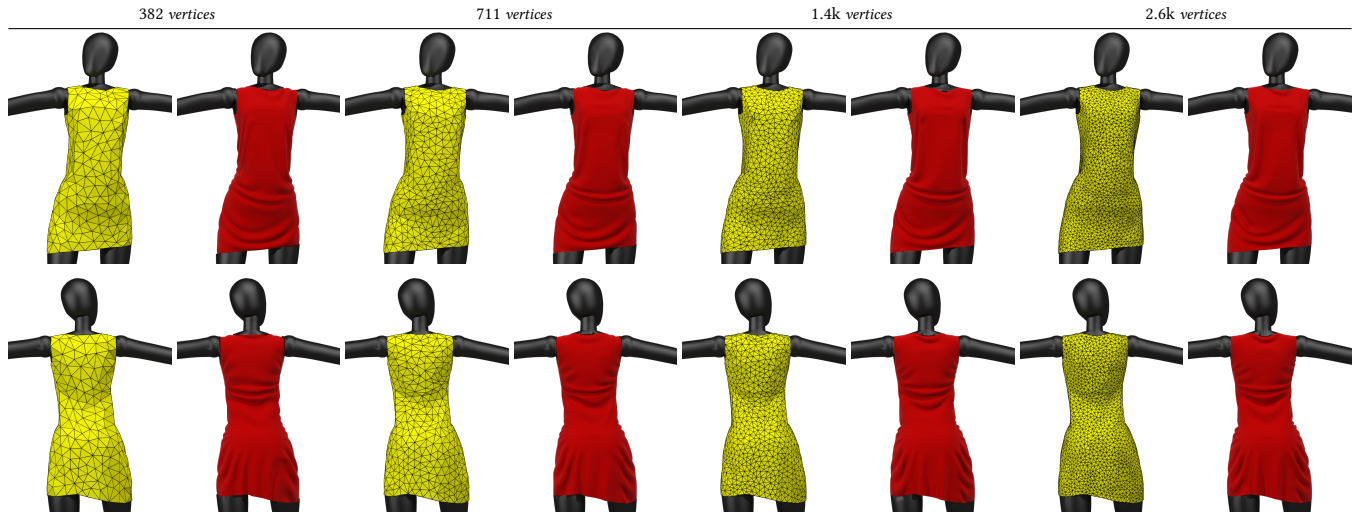


Fig. 20. Simulations of a draped dress with different mesh resolutions using our *TFW* model. The wireframe figures to the left of each image pair show the base mesh; the red meshes are the final results. Notice that *TFW* yields a consistent wrinkle pattern when the resolution reaches around 1.4k vertices.

- Jorge J. Moré and David J. Thuente. 1994. Line Search Algorithms with Guaranteed Sufficient Decrease. *ACM Trans. Math. Softw.* 20, 3 (Sept. 1994), 286–307. <https://doi.org/10.1145/192115.192132>
- Matthias Müller and Nuttapong Chentanez. 2010. Wrinkle Meshes. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Madrid, Spain) (SCA '10), 85–92. <https://doi.org/10.2312/SCA/SCA10/085-091>
- Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. 2007. Position based dynamics. *Journal of Visual Communication and Image Representation* 18, 2 (2007), 109 – 118. <https://doi.org/10.1016/j.jvcir.2007.01.005>
- Rahul Narain, Tobias Pfaff, and James F. O'Brien. 2013. Folding and Crumpling Adaptive Sheets. *ACM Trans. Graph.* 32, 4, Article 51 (July 2013), 8 pages. <https://doi.org/10.1145/2461912.2462010>
- Rahul Narain, Armin Samii, and James F. O'Brien. 2012. Adaptive Anisotropic Remeshing for Cloth Simulation. *ACM Trans. Graph.* 31, 6, Article 152 (Nov. 2012), 10 pages. <https://doi.org/10.1145/2366145.2366171>
- Igor Pak and Jean-Marc Schlenker. 2010. Profiles of Inflated Surfaces. *Journal of Nonlinear Mathematical Physics* 17, 02 (2010), 145–157. <https://doi.org/10.1142/S140292511000057X>
- Daniele Panozzo, E. Puppo, and Luigi Rocca. 2010. Efficient multi-scale curvature and crease estimation. *2nd International Workshop on Computer Graphics, Computer Vision and Mathematics, GraVisMa 2010 - Workshop Proceedings* (01 2010), 9–16.
- Joseph D. Paulsen, Evan Hohlfeld, Hunter King, Jiangshui Huang, Zhanlong Qiu, Thomas P. Russell, Narayanan Menon, Dominic Vella, and Benny Davidovitch. 2016. Curvature-induced stiffness and the spatial variation of wavelength in wrinkled sheets. *Proceedings of the National Academy of Sciences* 113, 5 (2016), 1144–1149. <https://doi.org/10.1073/pnas.1521520113>
- William H. Paulsen. 1994. What Is the Shape of a Mylar Balloon? *The American Mathematical Monthly* 101, 10 (1994), 953–958. <https://doi.org/10.2307/2975161>
- Allen C. Pipkin. 1986. The Relaxed Energy Density for Isotropic Elastic Membranes. *IMA Journal of Applied Mathematics* 36, 1 (01 1986), 85–99. <https://doi.org/10.1093/imamat/36.1.85>
- Allen C. Pipkin. 1994. Relaxed energy densities for large deformations of membranes. *IMA Journal of Applied Mathematics* 52, 3 (09 1994), 297–308. <https://doi.org/10.1093/imamat/52.3.297>
- Olivier Rémillard and Paul G. Kry. 2013. Embedded thin shells for wrinkle simulation. *ACM Trans. Graph.* 32, Article 82 (July 2013), 8 pages. Issue 4. <https://doi.org/10.1145/2461912.2462018>
- Damien Rohmer, Tiberiu Popa, Marie-Paule Cani, Stefanie Hahmann, and Alla Sheffer. 2010. Animation Wrinkling: Augmenting Coarse Cloth Simulations with Realistic-Looking Wrinkles. *ACM Transaction on Graphics* 29 (12 2010). <https://doi.org/10.1145/1866158.1866183>
- Igor Santesteban, Miguel A. Otaduy, and Dan Casas. 2019. Learning-Based Animation of Clothing for Virtual Try-On. *Computer Graphics Forum* 38, 2 (2019), 355–366. <https://doi.org/10.1111/cgfi.13643>
- Martin Seiler, Jonas Spillmann, and Matthias Harders. 2012. Enriching Coarse Interactive Elastic Objects with High-Resolution Data-Driven Deformations. In *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*. <https://doi.org/10.2312/SCA/SCA12/009-017>
- Andrew Selle, Jonathan Su, Geoffrey Irving, and Ronald Fedkiw. 2009. Robust High-Resolution Cloth Using Parallelism, History-Based Collisions, and Accurate Friction. *IEEE Transactions on Visualization and Computer Graphics* 15 (2009), 339–350. <https://doi.org/10.1109/TVCG.2008.79>
- Jonathan Richard Shewchuk. 1996. Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. In *Applied Computational Geometry Towards Geometric Engineering*. 203–222. <https://doi.org/10.1007/BFb0014497>
- Melina Skouras, Bernhard Thomaszewski, Peter Kaufmann, Akash Garg, Bernd Bickel, Eitan Grinspun, and Markus Gross. 2014. Designing Inflatable Structures. *ACM Transactions on Graphics* 33 (07 2014), 1–10. <https://doi.org/10.1145/2601097.2601166>
- Georg Sperl, Rahul Narain, and Chris Wojtan. 2020. Homogenized Yarn-Level Cloth. *ACM Trans. Graph.* 39, 4, Article 48 (July 2020), 16 pages. <https://doi.org/10.1145/3386569.3392412>
- David Steigmann. 1990. Tension-Field Theory. *Royal Society of London Proceedings Series A* 429 (05 1990), 141–173. <https://doi.org/10.1098/rspa.1990.0055>
- Rasmus Tamstorf, Toby Jones, and Stephen F. McCormick. 2015. Smoothed Aggregation Multigrid for Cloth Simulation. *ACM Trans. Graph.* 34, 6, Article 245 (Oct. 2015), 13 pages. <https://doi.org/10.1145/2816795.2818081>
- Wim M. van Rees, Etienne Vouga, and L. Mahadevan. 2017. Growth patterns for shape-shifting elastic bilayers. *Proceedings of the National Academy of Sciences* 114, 44 (2017), 11597–11602. <https://doi.org/10.1073/pnas.1709025114>
- Hugues Vandeparre, Miguel Piñeirua, Fabian Brau, Benoit Roman, José Bico, Cyprien Gay, Wenzhong Bao, Chun Ning Lau, Pedro M. Reis, and Pascal Damman. 2011. Wrinkling Hierarchy in Constrained Thin Sheets from Suspended Graphene to Curtains. *Physical Review Letters* 106, 22 (Jun 2011).
- Roman Vetter, Norbert Stoop, Falk K. Wittel, and Hans J. Herrmann. 2014. Simulating Thin Sheets: Buckling, Wrinkling, Folding and Growth. *Journal of Physics: Conference Series* 487 (mar 2014). <https://doi.org/10.1088/1742-6596/487/1/012012>
- Huamin Wang, Florian Hecht, Ravi Ramamoorthi, and James F. O'Brien. 2010. Example-Based Wrinkle Synthesis for Clothing Animation. *ACM Trans. Graph.* 29, 4, Article 107 (July 2010), 8 pages. <https://doi.org/10.1145/1778765.1778844>
- Ting Wang, Fu Chenbo, Fan Xu, Yongzhong Huo, and Michel Potier-Ferry. 2018. On the wrinkling and restabilization of highly stretched sheets. *International Journal of Engineering Science* 136 (12 2018), 1–16. <https://doi.org/10.1016/j.ijengsci.2018.12.002>
- Yu Wang, Alec Jacobson, Jernej Barbič, and Ladislav Kavan. 2015. Linear Subspace Design for Real-Time Shape Deformation. *ACM Trans. Graph.* 34, 4, Article 57 (July 2015), 11 pages. <https://doi.org/10.1145/2766952>
- Clarisse Weischedel. 2012. *A discrete geometric view on shear-deformable shell models*. PhD dissertation. Georg-August-Universität Göttingen.
- Wesley Wong and Sergio Pellegrino. 2006. Wrinkled membranes Part I: Experiments. *Journal of Mechanics of Materials and Structures - J MECH MATER STRUCT* 1 (05 2006), 3–25. <https://doi.org/10.2140/jomms.2006.1.3>
- Linbo Zhang, Tao Cui, and Hui Liu. 2009. A set of symmetric quadrature rules on triangles and tetrahedra. *Journal of Computational Mathematics* 27 (01 2009), 89–96.

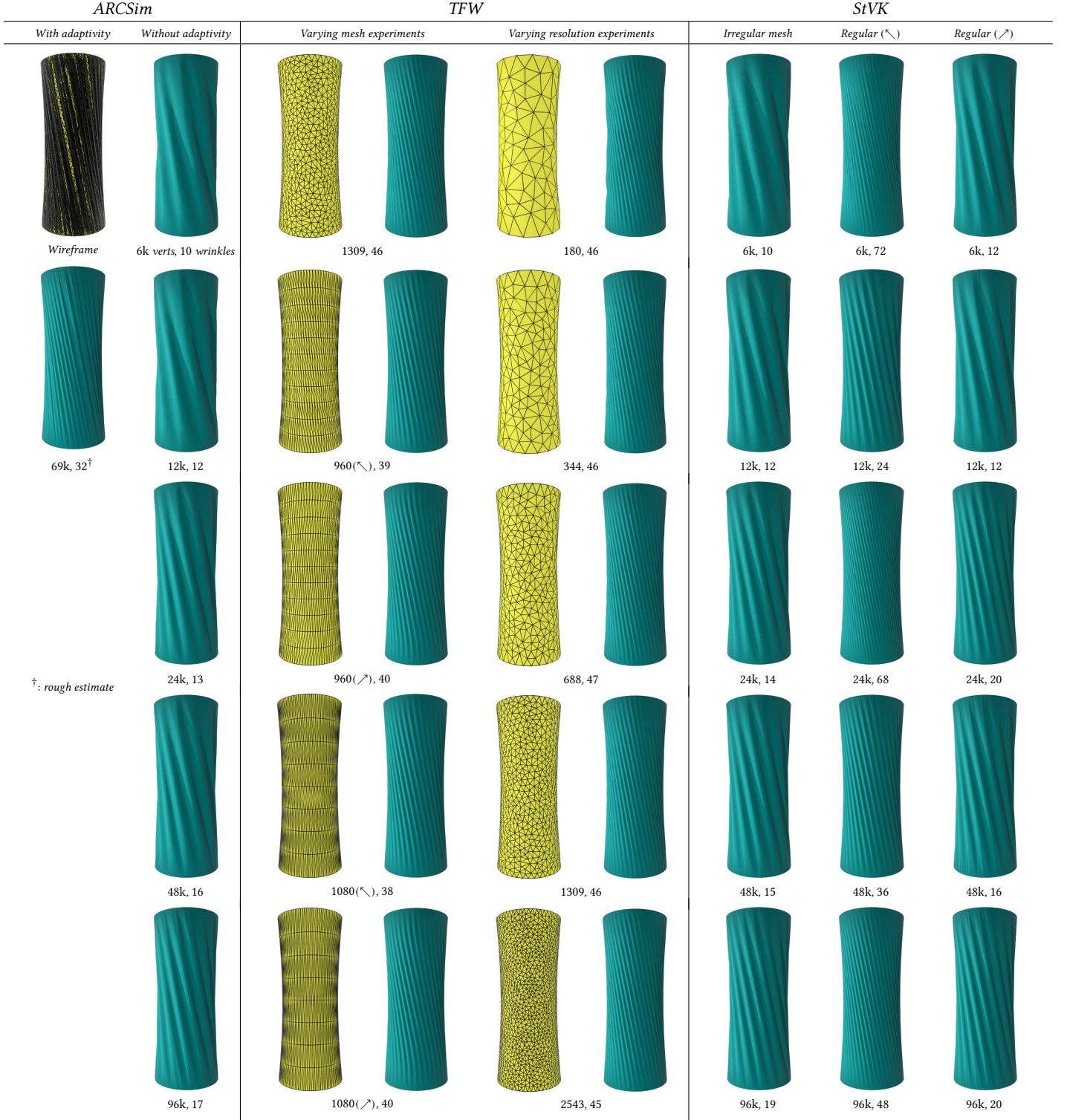


Fig. 21. Our results for the twisted cylinder experiments. Numbers below figures are (resolution, # wrinkles); for the adaptive ARCSim simulation, wrinkle count is a rough estimate as the irregularity of the wrinkles precludes an exact count. The symbols ( $\searrow$ ) and ( $\nearrow$ ) indicate use of a regular mesh with edges aligned along and against the wrinkles, respectively.



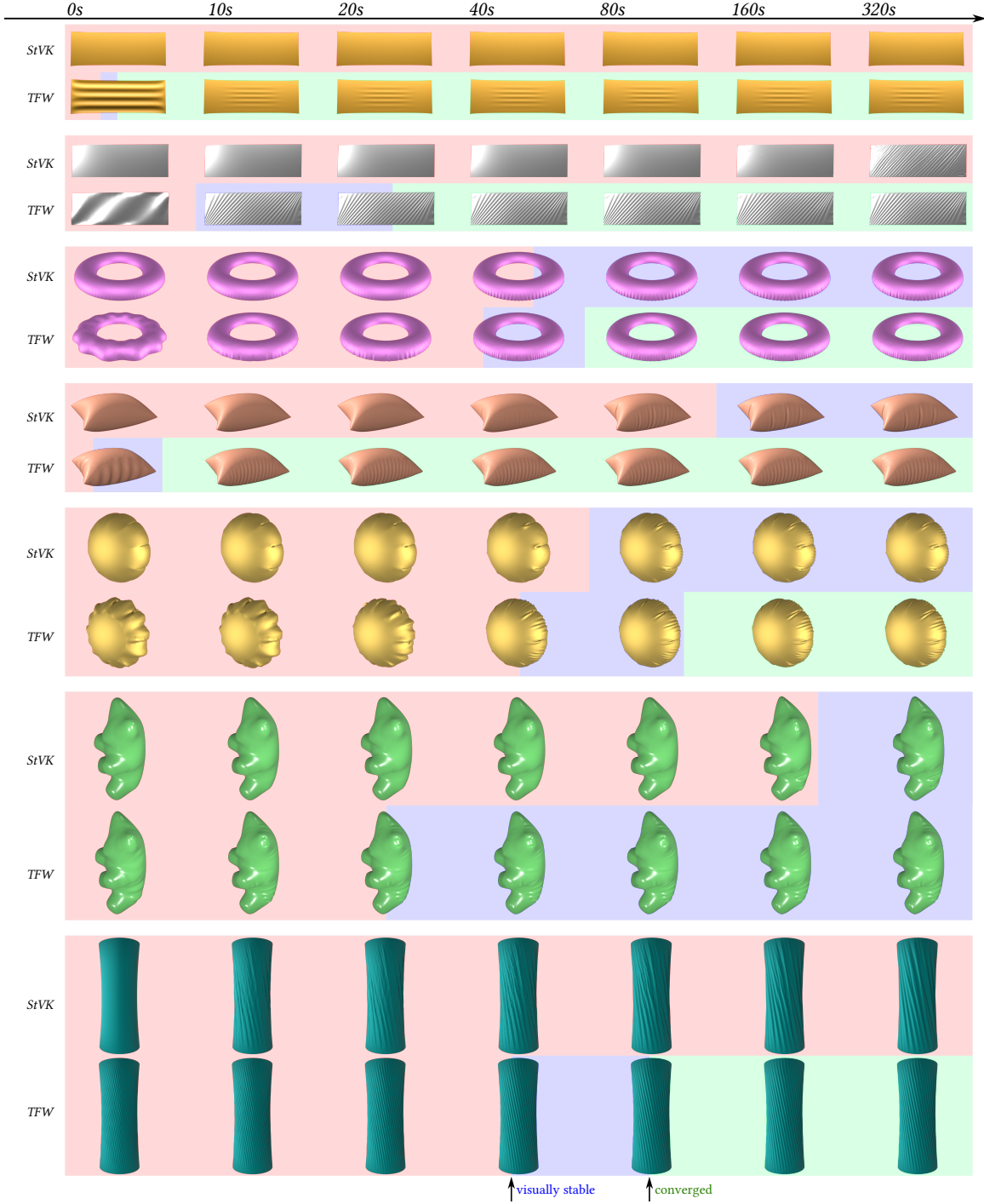


Fig. 22. Comparisons of our *TFW* algorithm and the baseline *StVK* simulations against wall clock time, on a log scale. Each rendered result is sampled at time corresponding to the image's left edge. Background color indicates current state of each simulation: light red indicates the simulation has not yet reached a visually-stable state, light blue indicates the simulation has become visually-stable, and light green means the solver has terminated due to having small stationarity residual. The boundary between colors correspond to the transitions between these three states, on the wall-clock-time axis. Note that the transitions between red and blue are exactly the times listed in Table 1.

## A KIRCHHOFF-LOVE SHELLS

$$\mathbf{s}(u, v, t) = \mathbf{r}(u, v) + t\hat{\mathbf{n}}(u, v)$$

where  $\hat{\mathbf{n}} = \frac{\mathbf{r}_u \times \mathbf{r}_v}{\|\mathbf{r}_u \times \mathbf{r}_v\|}$  is the unit normal of the midsurface. This map induces a (volumetric) metric  $\mathbf{g}$  on the slab  $U \times [-\frac{h}{2}, \frac{h}{2}]$ :

$$\mathbf{g} = \begin{bmatrix} \mathbf{I} - 2t\mathbf{\Pi} + O(t^2) & 0 \\ 0 & 1 \end{bmatrix}, \quad (31)$$

where

$$\mathbf{I} = d\mathbf{r}^T d\mathbf{r}, \quad \mathbf{\Pi} = -d\mathbf{r}^T d\hat{\mathbf{n}}$$

are the first and second fundamental forms of the midsurface.

If the residual strains in the shell are linear in the thickness direction, the shell's rest state can be recorded in terms of a "rest metric" with similar expression [Chen et al. 2018; van Rees et al. 2017]:

$$\bar{\mathbf{g}} = \begin{bmatrix} \mathbf{I}_u - 2t\mathbf{\Pi}_u & 0 \\ 0 & 1 \end{bmatrix}. \quad (32)$$

Notice that if we take the rest state as the parameter domain, which is almost always the case for sewn garments, then  $\mathbf{I}_u = \mathbf{id}$  and  $\mathbf{\Pi}_u = 0$ .

For a St. Venant-Kirchhoff material, and assuming in-plane strain is  $O(h)$ , the elastic energy in Equation (1) can be derived from the above setup, by integrating through the midsurface direction, and dropping energy terms of order higher than cubic [Weischedel 2012].

As mentioned in the main text, the matrix norm in the Koiter energy describes the elastic constitutive law,

$$\|\mathbf{M}\|_{\text{SV}}^2 = \frac{\alpha}{2} \text{tr}^2(\mathbf{M}) + \beta \text{tr}(\mathbf{M}^2),$$

where the Lamé parameters are related to the Young's modulus  $Y$  and Poisson's ratio  $\nu$  by

$$\alpha = \frac{Y\nu}{1-\nu^2}, \quad \beta = \frac{Y}{2(1+\nu)}. \quad (33)$$

We discretize the first fundamental form as piecewise constant, derived directly from the definition  $\mathbf{I} = d\mathbf{r}^T d\mathbf{r}$  and the fact that  $\mathbf{r}$  is linear within each triangle face  $\mathbf{f}_{ijk}$ :

$$\mathbf{I} = \begin{bmatrix} \|\mathbf{v}_j - \mathbf{v}_i\|^2 & (\mathbf{v}_j - \mathbf{v}_i) \cdot (\mathbf{v}_k - \mathbf{v}_i) \\ (\mathbf{v}_k - \mathbf{v}_i) \cdot (\mathbf{v}_j - \mathbf{v}_i) & \|\mathbf{v}_k - \mathbf{v}_i\|^2 \end{bmatrix},$$

Here the first fundamental form is expressed in the triangle's own barycentric coordinates.

For the second fundamental form, we follow the discretization of Grinspun [2003] and others [Chen et al. 2018; Weischedel 2012] based on jumps in the mid-edge normal:

$$\mathbf{\Pi}_b = 2 \begin{bmatrix} (\mathbf{v}_j - \mathbf{v}_i) \cdot (\hat{\mathbf{n}}_j - \hat{\mathbf{n}}_i) & (\mathbf{v}_k - \mathbf{v}_i) \cdot (\hat{\mathbf{n}}_j - \hat{\mathbf{n}}_i) \\ (\mathbf{v}_k - \mathbf{v}_i) \cdot (\hat{\mathbf{n}}_j - \hat{\mathbf{n}}_i) & (\mathbf{v}_k - \mathbf{v}_i) \cdot (\hat{\mathbf{n}}_k - \hat{\mathbf{n}}_i) \end{bmatrix},$$

where  $\hat{\mathbf{n}}_i$  is the mid-edge normal on the edge  $\mathbf{e}_{jk}$  opposite to  $\mathbf{v}_i$  on face  $\mathbf{f}_{ijk}$ . We take this mid-edge normal to be

- the unit face normal if  $\mathbf{e}_{jk}$  is a boundary edge;
- the average of the face normals on the two adjacent faces of  $\mathbf{e}_{jk}$ , otherwise.

## B TENSION FIELD THEORY

Given a parameter domain  $U \subset \mathbb{R}^2$  with material metric  $\mathbf{I}_u$  as described in Appendix A, the stretching energy density  $W_s$  in Equation (2) can be written as:

$$\begin{aligned} W_s(\lambda_1, \lambda_2) &= \|\mathbf{I}_u^{-1} \mathbf{I} - \mathbf{id}\|_{\text{SV}}^2 \\ &= \frac{\alpha}{2} (\lambda_1 + \lambda_2)^2 + \beta (\lambda_1^2 + \lambda_2^2). \end{aligned} \quad (34)$$

Here  $\lambda_{1,2}$  are the eigenvalues of the Green strain  $\mathbf{I}_u^{-1} \mathbf{I} - \mathbf{id}$ . Without loss of generality, we assume  $\lambda_1 \geq \lambda_2$ .

In tension field theory, this stretching energy density is modified to be identical to  $W_s$  in regions of pure tension, but so that the material exerts no force resisting compressive stress. In terms of the principal strains, this behavior is captured by the formula

$$\tilde{W}_s(\lambda_1, \lambda_2) = \begin{cases} 0, & \lambda_1 < 0, \lambda_2 < 0 \\ W_s(\lambda_1, \hat{\lambda}_2(\lambda_1)), & \lambda_1 \geq 0, \lambda_2 < \hat{\lambda}_2(\lambda_1) \\ W_s(\lambda_1, \lambda_2), & \lambda_1 \geq 0, \lambda_2 \geq \hat{\lambda}_2(\lambda_1), \end{cases}$$

where

$$\hat{\lambda}_2(\lambda_1) = \arg \min_{\lambda_2} W_s(\lambda_1, \lambda_2).$$

See Montes et al. [2020] for the full derivation of these expressions.

## C IN-PLANE WRINKLE CORRECTION CALCULATION

In this section, we use the notation and definitions of Section 3.4 for  $\mathbf{w}, \mathbf{w}^\perp$  and  $d\phi^\perp$ , and derive the expressions for  $\mathbf{v}_1$  and  $\mathbf{v}_2$  cited in the main paper.

First, notice that we can zero out the term (15) by simply setting

$$\mathbf{v}_2 = \frac{a^2}{8} \mathbf{I}_b^{-1} d\phi^T.$$

The  $\mathbf{v}_1$  term is more involved. Writing  $\mathbf{X} = -2a\mathbf{\Pi}_b$  and  $\mathbf{y} = \mathbf{I}_b \mathbf{v}_1$ , and using the notation  $[\mathbf{M}]_T = \mathbf{M} + \mathbf{M}^T$ , minimizing the term (14) amounts to solving:

$$\min_{\mathbf{v}_1 \in \mathbb{R}^2} \left\| \mathbf{I}_u^{-1} \left( \mathbf{X} + [\mathbf{y} d\phi^T]_T \right) \right\|_{\text{SV}}^2 \quad (35)$$

One can check that, for any symmetric matrix  $\mathbf{M}$ ,

$$\begin{aligned} \|\mathbf{I}_u^{-1} \mathbf{M}\|_{\text{SV}}^2 &= \frac{\alpha}{2} \left( \|\hat{\mathbf{w}}\|_M^2 + \|\hat{\mathbf{w}}^\perp\|_M^2 \right) \\ &\quad + \beta \left( \|\hat{\mathbf{w}}\|_M^2 + 2 \left( \hat{\mathbf{w}}^T \mathbf{M} \hat{\mathbf{w}}^\perp \right)^2 + \|\hat{\mathbf{w}}^\perp\|_M^2 \right) \end{aligned} \quad (36)$$

using the fact

$$\text{tr}(\mathbf{I}_u^{-1} \mathbf{M}) = \hat{\mathbf{w}}^T \mathbf{M} \hat{\mathbf{w}} + (\hat{\mathbf{w}}^\perp)^T \mathbf{M} \hat{\mathbf{w}}^\perp$$

where  $\hat{\mathbf{w}}$  and  $\hat{\mathbf{w}}^\perp$  are the normalized  $\mathbf{w}$  and  $\mathbf{w}^\perp$  under  $\mathbf{I}_u$  norm. By setting (recall:  $\mathbf{w} = \mathbf{I}_u^{-1} d\phi^T$ )

$$\tilde{\mathbf{y}} = \|\mathbf{w}\|_{\mathbf{I}_u} \mathbf{y}, \quad \mathbf{M} = \mathbf{X} + [\tilde{\mathbf{y}} \hat{\mathbf{w}}^T \mathbf{I}_u]_T$$

and representing  $\tilde{\mathbf{y}}$  in the basis  $\{\mathbf{I}_u \hat{\mathbf{w}}, \mathbf{I}_u \hat{\mathbf{w}}^\perp\}$ ,

$$\tilde{\mathbf{y}} = \tilde{x}_1 \mathbf{I}_u \hat{\mathbf{w}} + \tilde{x}_2 \mathbf{I}_u \hat{\mathbf{w}}^\perp,$$

Equation (35) can be converted into a quadratic problem

$$\min_{\tilde{x}_1, \tilde{x}_2} \frac{\alpha}{2} (c_1 + 2\tilde{x}_1 + c_2)^2 + \beta \left( (c_1 + 2\tilde{x}_1)^2 + c_2^2 + 2(c_3 + \tilde{x}_2)^2 \right)$$

where  $c_1 = \hat{\mathbf{w}}^T X \hat{\mathbf{w}}$ ,  $c_2 = (\hat{\mathbf{w}}^\perp)^T X \hat{\mathbf{w}}^\perp$  and  $c_3 = \hat{\mathbf{w}}^T X \hat{\mathbf{w}}^\perp$ . The optimal solution is given by

$$\begin{aligned}\tilde{x}_1^* &= -\frac{1}{2} \left( c_1 + \frac{\alpha}{\alpha + 2\beta} c_2 \right) = -\frac{1}{2} \frac{\mathbf{w}^T X \mathbf{w}}{\mathbf{w}^T \mathbf{I}_u \mathbf{w}} - \frac{\alpha}{2\alpha + 4\beta} \frac{(\mathbf{w}^\perp)^T X \mathbf{w}^\perp}{(\mathbf{w}^\perp)^T \mathbf{I}_u \mathbf{w}^\perp} \\ \tilde{x}_2^* &= -c_3 = -\frac{\mathbf{w}^T X \mathbf{w}^\perp}{\|\mathbf{w}\|_{\mathbf{I}_u} \|\mathbf{w}^\perp\|_{\mathbf{I}_u}} \\ \mathbf{y}^* &= -\left( \frac{\alpha}{2\alpha + 4\beta} \frac{\text{tr}(\mathbf{I}_u^{-1} X)}{\|\mathbf{w}\|_{\mathbf{I}_u}^2} + \frac{\beta}{\alpha + 2\beta} \frac{\mathbf{w}^T X \mathbf{w}}{\|\mathbf{w}\|_{\mathbf{I}_u}^4} \right) d\phi^T \\ &\quad - \frac{\mathbf{w}^T X \mathbf{w}^\perp}{\|\mathbf{w}\|_{\mathbf{I}_u}^2 \|\mathbf{w}^\perp\|_{\mathbf{I}_u}^2} (d\phi^\perp)^T\end{aligned}$$

and the optimal value attained is

$$\frac{\beta(\alpha + \beta)}{(\alpha/2 + \beta)} c_2^2 = \frac{\beta(\alpha + \beta)}{(\alpha/2 + \beta)} \frac{\left[ (\mathbf{w}^\perp)^T X \mathbf{w}^\perp \right]^2}{\|\mathbf{w}^\perp\|_{\mathbf{I}_u}^4}.$$

Unwinding the changes of variables leads to the formula for  $\mathbf{v}_1$  in Section 3.4.

## D DERIVATION OF STRETCHING TERM IN *TFW* MODEL

In Equation (19), we have

$$\mathbf{r} \approx \mathbf{r}_b + d\mathbf{r}_b \mathbf{I}_b^{-1} \left( a \sin \phi \mathbf{v} + \frac{a^2}{8} \sin 2\phi d\phi^T \right) + a \cos \phi \hat{\mathbf{n}}_b.$$

Applying the assumptions about fast and slow variables stated in Section 3.3, we can compute  $d\mathbf{r}$  in the following way:

$$\begin{aligned}d\mathbf{r} &= d\mathbf{r}_b \left( \mathbf{id} - a \cos \phi \mathbf{I}_b^{-1} \mathbf{I}_b + \sin \phi \left( \mathbf{I}_b^{-1} \mathbf{v} \right) da + a \cos \phi \left( \mathbf{I}_b^{-1} \mathbf{v} \right) d\phi \right. \\ &\quad \left. + \frac{a}{4} \sin 2\phi \left( \mathbf{I}_b^{-1} d\phi^T \right) da + \frac{a^2}{4} \cos 2\phi \left( \mathbf{I}_b^{-1} d\phi^T \right) d\phi \right) \\ &\quad + \hat{\mathbf{n}}_b \left( \cos \phi da - a \sin \phi d\phi + a \sin \phi \left( \mathbf{I}_b^{-1} \mathbf{v} \right)^T \mathbf{I}_b \right. \\ &\quad \left. + \frac{a^2}{8} \sin 2\phi \left( d\phi \mathbf{I}_b^{-1} \right) \mathbf{I}_b \right) \\ &= d\mathbf{r}_b (\mathbf{id} + A) + \hat{\mathbf{n}}_b B,\end{aligned}$$

where the matrix  $A$  and vector  $B$  collect the various terms in the above expression. We then compute

$$\begin{aligned}(\mathbf{id} + A)^T \mathbf{I}_b (\mathbf{id} + A) &= \mathbf{I}_b + [\mathbf{I}_b A]_T + o(A) \\ &\approx \mathbf{I}_b - 2a \cos \phi \mathbf{I}_b + \sin \phi [\mathbf{v} da] + a \cos \phi [\mathbf{v} d\phi]_T \\ &\quad + \frac{a}{4} \sin 2\phi [d\phi^T da]_T + \frac{a^2}{2} \cos 2\phi d\phi^T d\phi \\ &= \mathbf{I}_b + \frac{a}{4} \sin 2\phi [d\phi^T da]_T + \frac{a^2}{2} \cos 2\phi d\phi^T d\phi \\ &\quad - 2a \cos \phi \mathbf{I}_b + a \cos \phi [\mathbf{v} d\phi]_T + o(\|da\|)\end{aligned}$$

and

$$\begin{aligned}B^T B &= \cos^2 \phi da^T da + a^2 \sin^2 \phi d\phi^T d\phi - a \sin \phi \cos \phi [d\phi^T da] \\ &\quad + o\left(a^2 \|d\phi\|^2\right) + o(a\|\mathbf{v}\|) + o(a\|d\phi^T da\|) \\ &\approx \cos^2 \phi da^T da + a^2 \sin^2 \phi d\phi^T d\phi - a \sin \phi \cos \phi [d\phi^T da].\end{aligned}$$

To avoid (even more) clutter, implicit in the above expressions is the use of one-form and vector norms  $\mathbf{I}_u^{-1}$  and  $\mathbf{I}_u$ , respectively, and the matrix norm  $\|\cdot\|_{\text{SV}}$ .

Then, after dropping the high order terms, we get

$$\begin{aligned}W_s &= \mathbf{I}_u^{-1} (d\mathbf{r} d\mathbf{r} - \mathbf{I}_u) \\ &= \mathbf{I}_u^{-1} (\mathbf{I}_b - \mathbf{I}_u + \frac{1}{2} da^T da + \frac{1}{2} a^2 d\phi^T d\phi) \\ &\quad + \cos \phi \left( \mathbf{I}_u^{-1} (-2a \mathbf{I}_b + a [\mathbf{v} d\phi]_T) \right) \\ &\quad + \sin 2\phi \left( \mathbf{I}_u^{-1} \left( -\frac{a}{4} [d\phi^T da]_T \right) + \cos 2\phi \left( \frac{1}{2} \mathbf{I}_u^{-1} da^T da \right) \right).\end{aligned}$$

From here we recover the expression for the wrinkle field stretching energy in Section 3.5 by plugging in the definition of  $\mathbf{v}$ , and applying the coarse-graining operator.

## E ADDITIONAL IMPLEMENTATION DETAILS

In this appendix, we flesh out some of the steps described at a high level in Section 5 of the main paper.

### E.1 Triangulation

We Delaunay-triangulate the parameter domain  $U$  to create a simulation mesh. In many of our examples, the parameter domain is given as a set of disconnected patches, which are to be sewn together into a garment or balloon along shared boundaries; we do so to generate a single connected simulation mesh  $K$ .

In Section 6.3, we analyze the effect of meshing on the *TFW* results. We observe that although our method is fairly robust to mesh resolution and tessellation, using a highly symmetric mesh, or one whose edge directions have consistent bias, can result in artifacts, both during simulation of *TFT* and during Loop subdivision. We therefore recommend always using an irregular but coarse Delaunay mesh.

### E.2 Initialization

In this step, we choose initial guesses for  $a$  and  $\omega$ , based on the strain of the base mesh embedding  $V_b$ . We assume each triangle  $f$  in  $F$  has constant strain (see Appendix A for details on how we discretize strain and related quantities), and compute the direction  $\mathbf{w}$  and magnitude  $\epsilon_{\mathbf{w}}$  of the most negative principal strain. We initialize amplitude to  $k\sqrt{2\epsilon_{\mathbf{w}}}$ , where  $k$  is an arbitrary constant (in our experiments, this constant is set to  $\min[0.01, 0.1 \cdot \text{bbox}(V_b)]$ , where  $\text{bbox}(V_b)$  is the diameter of the bounding box around the base mesh), and compute a target frequency  $\omega_e$  for each edge  $e$  of  $f$  using Equation (8). (If the triangle has no negative principal strain, we use  $a = 0$  and  $\omega_e = 0$  instead.)

Note that this procedure does not generally yield an integrable one-form  $\omega$  (in fact, the two triangles neighboring  $e$  usually will

not even agree on  $\omega_e$ ). We project to the closest curl-free one-form  $\omega$  in the least-squares sense using the Helmholtz decomposition.

### E.3 Amplitude and Phase Optimization Details

Recall from Section 5 that we discretize the parameter domain  $U$  by a coarse mesh  $K = (V_u, F, E)$ , and discretize  $a$  as a piecewise-linear function on  $K$  and  $\phi$  as a one-form on the edges  $E$ . We can then write the objective function in Equation (30) as a sum of contributions from triangles not in  $W$ :

$$E^{\text{wf}} = \frac{1}{2} \sum_{f \in F, f \notin W} \int_f \left( \frac{h}{4} W_s^{\text{wf}} + \frac{h^3}{12} W_b^{\text{wf}} \right) \sqrt{\det I_u} dA, \quad (37)$$

where  $W_s^{\text{wf}}$  and  $W_b^{\text{wf}}$  collect the terms in Equations (20) and (26).  $da$  is constant over each triangle, as are the fundamental forms  $I_u, II_u$ . The other terms we need in order to implement Equation (37) are  $\kappa_\perp$  and  $d\phi$ .

To estimate curvature of the wrinkle crests, we first compute a vector  $\mathbf{w}$  per triangle, as described in Section 4.2, based on that triangle's (constant) base mesh strain tensor. We then robustly estimate principal curvatures and curvature directions via quadratic fitting [Panozzo et al. 2010], which allows us to compute  $\kappa_\perp$ . Note that  $\kappa_\perp$  is a constant: it does not vary over the course of optimizing  $E^{\text{wf}}$ .

Integrability induces one linear equality constraint for each triangle not in  $W$ , enforcing that the circulation  $d\omega$  of  $\omega$  around the face is zero:

$$\omega_{ij} + \omega_{jk} + \omega_{ki} = 0 \quad (38)$$

for each face  $f_{ijk} \notin W$  with edges  $\{\mathbf{e}_{ij}, \mathbf{e}_{jk}, \mathbf{e}_{ki}\}$ .

On faces not in  $W$ , integrability allows us to recover a constant  $d\phi$  on that face from the values of  $\omega$  on the face's edges, from the defining equations

$$d\phi(\mathbf{v}_j - \mathbf{v}_i) = \omega_{ij}, \quad d\phi(\mathbf{v}_k - \mathbf{v}_j) = \omega_{jk}, \quad d\phi(\mathbf{v}_i - \mathbf{v}_k) = \omega_{ki}.$$

The integrability constraint on  $\omega$  is exactly the condition that enforces that this overconstrained system of equations has a solution (in practice we compute  $d\phi$  using only the first two equations, and discarding the third as redundant).

Forming a global vector  $\mathbf{x} \in \mathbb{R}^{|V_b|+|E|}$  by concatenating the unknowns  $a$  and  $\omega$ , applying the above discretization and three-point quadrature [Zhang et al. 2009] to compute the integrals in Equation (37) yields a degree-eight polynomial function  $E(\mathbf{x})$  discretizing  $E^{\text{wf}}$ . We minimize this polynomial subject to the simple non-negativity constraints on amplitudes and the curl constraint,  $C\mathbf{x} = 0$ , on non- $W$  faces,

$$\min_{\mathbf{x}} E(\mathbf{x}) \quad \text{s.t.} \quad S\mathbf{x} \geq 0, C\mathbf{x} = 0 \quad (39)$$

where  $S$  is the selector matrix extracting amplitudes from  $\mathbf{x}$ . We solve this system via sequential quadratic programming. At each iteration, we use the NASOQ QP solver [Cheshmi et al. 2020] to compute a descent directions  $\delta\mathbf{x}$ , by solving the sparse, linearly constrained quadratic minimization problem

$$\min_{\delta\mathbf{x}} \frac{1}{2} \delta\mathbf{x}^T H \delta\mathbf{x} \quad \text{s.t.} \quad S[\mathbf{x}_k + \delta\mathbf{x}] \geq 0, C\delta\mathbf{x} = 0$$

where  $\mathbf{x}_k$  is the current iterate of  $\mathbf{x}$  and  $H$  is a convexification of the energy Hessian  $HE(\mathbf{x}_k)$  (see below). We perform a line search [Moré

and Thuente 1994] in the  $\delta\mathbf{x}$  direction to ensure each SQP iteration decreases the energy and does not violate the inequality constraints. We terminate this optimization process when one of following termination criteria are satisfied: (1) change in energy is smaller than  $10^{-10}$ , (2) the stationarity residual of (39) is smaller than  $10^{-6}$ , (3) the update to  $\mathbf{x}$  is smaller than  $10^{-10}$ , (4) reach the maximum iteration steps (1000 by default). We observe that, as expected, maximum (unit-length) step sizes are accepted near optimality and so always generate a feasible solution satisfying the constraints.

*Hessian Projection.* The quadratic form  $H$  must be positive-definite in order for the above SQP scheme to succeed, since otherwise the search direction  $\delta\mathbf{x}$  cannot be guaranteed to be a descent direction. We therefore select  $H$  using one of two methods that ensure it is positive-definite:

- computing the Hessian of each triangle's contribution to the sum in Equation (37), projecting that local Hessian to the closest positive-definite matrix (using SVD), and then summing those projected local Hessians to yield  $H$ ;
- setting  $H = HE(\mathbf{x}_k) + \epsilon \mathbf{I}$ , where  $\epsilon$  is a constant larger than the most-negative eigenvalue in  $HE(\mathbf{x}_k)$ , found via binary search.

We leave further research into methods for projecting the energy Hessian, or for combining the existing approaches into a high-performance metastrategy, to future work; for the results shown in this paper we use the first method at the beginning of the optimization, and switch to the second once the energy decrease per step becomes smaller than  $10^{-8}$ . See Section F.3 for convergence plots of the SQP and additional discussion.

### E.4 Phase Field Extraction

To visualize the wrinkled surface, we need to convert  $\omega$  back into a phase field  $\phi$ . Although the curl constraints ensure that  $\omega$  is *locally* integrable, there is no guarantee that a  $\phi$  globally exists with  $d\phi = \omega$ . In the sphere drape example, for instance, it is possible that the optimal  $\omega$  encodes a fractional number of wrinkles around the cloth circumference.

We borrow from the parameterization literature [Bommes et al. 2009] the idea of *rounding*  $\omega$  to the nearest  $\phi$ : we take  $K$ , remove the wrinkle-free faces  $F$ , and cut the result into a topological disk. We use Gurobi [2020] to solve

$$\min_{\phi} \|d\phi - \omega\|^2$$

subject to the constraint that the jump at each cut edge is an integer multiple of  $2\pi$ .

The resulting  $\phi$  is defined on a triangle soup made from  $F$ ; i.e. two neighboring faces on  $K$  that were cut along their common edge might disagree on the value of  $\phi$  at their shared vertices. But since this disagreement is always a multiple of  $2\pi$ , the cuts are invisible during visualization.

### E.5 Upsampling and Visualization

The output of the above *TFW* pipeline is the very coarse base mesh, and the wrinkle field  $(a, \phi)$  defined on its vertices. To visualize the final wrinkled shell, the mesh and wrinkle fields must be upsampled

(note that directly displacing the vertices of  $K$  according to the wrinkle field is not useful, as a single triangle often hosts multiple wavelengths of wrinkles). We explicitly materialize an upsampled mesh by applying Loop subdivision on  $K$  (although, in principle, the visualization could alternately be done on the fly with a tessellation shader) and also applying the subdivision stencil to  $a$  and  $\phi$ . Some care is needed when subdividing  $\phi$  to correctly account for: (1) the integer period jumps that can occur in  $\phi$  across neighboring triangles, and (2) triangles  $W$  where  $\phi$  is missing.

We then displace the vertices of the upsampled base mesh using Equation (19). Whereas the  $v_1$  term in this equation is crucial to the correct physical modeling of the elastic energy landscape, we find that this term has only a very slight effect on the visual appearance of the wrinkled surface during the upsampling. We thus drop the  $v_1$  term from just this final visualization (but not from prior computation of the  $TFW$  model) when upsampling and visualizing all examples shown in this paper. The  $v_1$  term is nontrivial to estimate on the upsampled base mesh, and subject to noise in regions where  $d\phi$  is small. By contrast, the  $v_2$  term is included in this visualization step as it is critical throughout. If we were to omit the  $v_2$  in-plane term we would obtain unnatural-looking undulations (see Figure 5, right, for an example) rather than the natural-looking, "bulging" wrinkles with clear overhangs between wrinkles seen in Figure 5, left.

## F ADDITIONAL PERFORMANCE EXPERIMENTS AND DATA

In this Appendix, we provide more detailed data and discussion related to the performance experiments reported in Section 6.5.

### F.1 Per-Iteration Timing Breakdown

We instrumented the wall-clock time required by each component of one optimization step of  $StVK$  and  $TFW$ , and report these timings (averaged over experiments and iterations) in Figure 23. For both methods, the base solver (QP for  $TFW$  and linear for  $StVK$ ) takes the majority of time ( $\approx 70\%$  for  $StVK$  and  $\approx 55\%$  for  $TFW$ ). We made a best effort to optimize both the  $StVK$  and  $TFW$  code, and that the QP solver time dominates in both methods in Figure 23 confirms that there are no gross inefficiencies remaining in either implementation.

For  $StVK$ , we used parallel supernodal sparse Cholesky decomposition, provided by SuiteSparse [Chen et al. 2008], as the solver, and the computational expense of the solve is due to the large size of the hessian matrix. For  $TFW$ , we use NASOQ [Cheshmi et al. 2020] as our QP solver, and the bulk of the expense is due to the presence of the integrability equality constraints on  $\omega$  and the inequality constraints on  $a$ . NASOQ is not optimized for our (very simple) box inequality constraints; in Table 2, we list the average time per iteration spend by NASOQ, and compare to a baseline where we drop the constraints and use CHOLMOD to solve the  $TFW$  QP instead. The latter numbers are substantially faster than the former and give a sense of the performance ceiling for  $TFW$ , should NASOQ be replaced by a more performant QP solver.

Source code for our implementations of both  $TFW$  and  $StVK$  are available in the supplementary material.

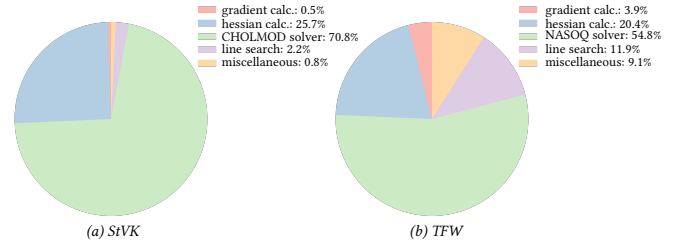


Fig. 23. Breakdown of average time required by each component of  $TFW$  and  $StVK$  during one optimization iteration (averaged over all iterations of all collision-free examples in the paper). See Tables 2 and 4 for additional timing breakdowns for  $TFW$  and  $StVK$ , respectively. “Miscellaneous” includes bookkeeping such as updating the state variables, printing information about the solver state to the console, checking for termination, etc.

### F.2 Residual Plots for Each Example

For each of our examples, we provide plots in Figure 24 of the gradient residual (for  $StVK$ ) and of the stationarity residual, i.e. gradient projected onto the constraint manifold using the current values of the Lagrange multipliers, for  $TFW$ . Both are plotted against wall clock time. We also show stills of each simulation at the chosen visually-stable time, as well as many iterations later, to illustrate that there is indeed no significant visual difference between the results at these two times.

For  $TFW$ , we see that the point at which a simulation becomes visually stable (chosen by inspection of the simulation output) approximately matches the onset of a plateau in the residual plot. The situation for  $StVK$  is less clear. In future work, it would be practically useful to formalize these observations into a quantitative termination condition corresponding to being “visually stable.”

### F.3 Convergence Discussion

In Tables 2 and 3, we provide additional timing data about termination of the simulations shown in Figure 22: recall that termination occurs on each timeline where the background color changes from blue to green, and that we terminate when either the residual infinity norm is smaller than  $10^{-6}$  or the optimization exceeds 1000 iterations. Note that the gradient of energy with respect to position has different units than the gradient with respect to amplitude  $a$  or to frequency  $\omega$ ; it is thus only meaningful to compare  $TFW$  results to each other and  $StVK$  results to each other, since each method is essentially using a different termination condition.

In Figure 24, we observe that the gradient norm eventually converges quadratically to zero, as expected. On the other hand, for some examples the  $TFW$  stationarity residual appears to converge only linearly. We believe the reason for this behavior is the non-negativity constraint on  $a$ : near optimality, the optimization problem in Equation (39) becomes convex, but *only in the feasible cone* delineated by the equality constraints and active inequality constraints. In particular, near optimality the unconstrained Hessian can be indefinite, even though the second-order change in energy is positive in every feasible direction. Unfortunately, standard QP solvers, including the ones we currently use (NASOQ) generally only accepts



positive quadratic forms, requiring us to project any indefinite Hessian to a nearby positive-definite matrix (see Appendix E for details). The use of this modified Hessian during SQP prevents quadratic convergence; in future work, NASOQ could potentially be replaced by a QP solver that does not require modifying the Hessian in cases where it is indefinite despite the constrained problem being locally convex.

#### F.4 Initialization of *StVK*

Like in all statics problems, our *StVK* optimization requires, and has performance sensitive to, an initial guess. In the main text, we propose using the *TFT* base mesh as the initial guess for *StVK*, both to maximize *StVK*'s performance, and for maintaining consistency of the experimental setup with *TFW*. We performed experiments to justify this choice, with results listed in Table 4. Instead of the *TFT* base mesh, in these experiments we used a problem-specific “flat” state as the initial guess: for the stretched sheet, sheared rectangle, torus, and balloon problems, we simply take the 2D rest mesh as the flat state. For the teabag we export an unwrinkled, pre-optimization initial guess from *Marvelous Designer*, and for the teddy, we start from the unwrinkled geometry provided by Skouras et al. [2014]. For the twisted cylinder problem, we use an untwisted cylinder as the flat state. Unsurprisingly, the *TFT* solution is universally a better initial guess than these alternate flat states, since the *TFT* solution is expected to match the *StVK* optimum, up to missing fine wrinkles.

#### G VIDEO COMPARISONS

In the supplementary materials, we provide videos of the equal-effort comparison experiments illustrated in Figure 22. We play back the optimization iterates for both *TFW* and *StVK*, where playback time is a multiple of wall clock time, chosen so that each clip

plays at reasonable speed. For *TFW*, we visualize amplitude  $a$  as a scalar color field on the base mesh (white is zero amplitude), and likewise draw  $I_u^{-1}\omega^T$  as a vector field on  $r_b$  (left animation). We show the wrinkled surface  $r_w$  as well (middle animation; note that since we only recover phase  $\phi$  up to an unknown global phase shift, which is not necessarily temporally coherent between solver iterations, the wrinkles on the surface can “drift” incoherently between iterations; see Section 7 for more discussion of the phase ambiguity). For *StVK*, we show the predicted vertex positions at each iteration (right animation).

We use the same background color as in Figure 22 to indicate the status of *TFW* and *StVK* at the each frame of the animations. We also show, at the bottom of the video, a wall-clock time axis for each animation, on which we mark the transition times where each simulation becomes visually stable or terminates. In most cases, termination of the *TFW* algorithm occurs well after *TFW* has reached a visually stable state, and well before *StVK* does so. We speed up the second portion of the animation (where *TFW* has terminated and *StVK* is still running) to keep the movie length reasonable—precise playback speed information is provided above the time axis. Each clip ends when the *StVK* simulation terminates, and we pause each animation for five seconds at the end.

Some notable behavior that we observe in the videos: for the *TFW* stretched sheet experiment, amplitude first vanishes globally over the sheet, and then wrinkles emerge at the center of the sheet, as predicted by theory. Notice also that in some examples (such as the Mylar balloon and torus), the vector field  $I_u^{-1}\omega^T$  has large magnitude near singularities. This behavior is expected (and corresponds to high-frequency, low amplitude wrinkling at points where wrinkles converge to a singular points).

Models	#vertices	<i>TFT</i>		<i>TFW</i>		Average time per iteration (s)		
		#iterations	total time (s)	#iterations	total time (s)	total	in NASOQ	CHOLMOD
sphere drape	1.6k	—	—	46	15.06	0.33	0.18	0.013
symmetric dress	1.4k	—	—	72	29.74	0.41	0.28	0.011
asymmetric dress	3.4k	—	—	50	38.47	0.77	0.56	0.030
pants	4k	—	—	74	141.74	1.92	1.52	0.029
stretched sheet	522	10	0.45	23	3.93	0.17	0.10	0.0039
sheared rectangle	1k	21	2.27	147	27.52	0.19	0.073	0.0082
torus	2k	11	2.27	125	74.82	0.60	0.44	0.017
balloon	5k	33	16.04	91	125.33	1.38	1.08	0.039
teabag	936	8	0.78	46	7.32	0.16	0.057	0.0034
teddy*	2k	3	0.93	1001	322.25	0.32	0.19	0.0063
twisted cylinder	688	12	0.69	813	104.92	0.13	0.048	0.0056

Table 2. Additional timing information for the *TFW* solver. The *TFT* columns list the number of iterations and wall clock time spent computing the base mesh (Section 4.1), and the *TFW* columns give the same information, for the solve for the wrinkle field variables (Sections 4.2 and 5.2). Note that the *TFW* times are for when we terminate the *TFW* simulation due to small stationarity residual, or maximum iterations, and do *not* correspond to termination at the visually-stable time (see Section F.3 for more discussion). Note that the time needed to compute the base mesh is negligible compared to the SQP solve for amplitude and frequency. The last three columns, from left to right, list: the average wall clock time required by one *TFW* iteration, the amount of that time spent specifically inside NASOQ (see also Figure 23), and the amount of time CHOLMOD would require for the same solve if the constraints were ignored. This latter number serves as a ceiling for how efficient *TFW* might be, if the NASOQ solver were replaced by another, more efficient code.

Models	#vertices	<i>TFT</i>		<i>StVK</i>		Average time per iteration (s)	
		#iterations	total time (s)	#iterations	total time (s)	total	in CHOLMOD
stretched sheet	260k	17	640.39	22	1178.55	53.57	42.63
sheared rectangle	20k	49	101.77	307	936.55	3.05	2.16
torus	40k	13	84.47	46	337.91	3.25	2.02
balloon	40k	291	1907.74	67	547.68	8.00	5.63
teabag	30k	11	34.11	98	521.08	5.32	3.59
teddy	98k	10	214.48	59	1188.25	20.14	14.44
twisted cylinder	960k	33	308.03	424	6071.73	14.32	9.80

Table 3. Additional timing information for the *StVK* solver. The *TFT* columns list the number of iterations and wall clock time spent computing the base mesh (Section 4.1), and the *StVK* columns give the same information, for the solve for the static shape using *TFT* as the initial guess. Note that the *StVK* times are for when we terminate the *StVK* simulation due to small gradient residual, or maximum iterations, and do *not* correspond to termination at the visually-stable time (see Section F.3 for more discussion). The last two columns list the average wall clock time required by one *StVK* iteration, and the amount of that time spent specifically inside CHOLMOD (see also Figure 23).

Models	#vertices	<i>StVK</i>		Average time per iteration (s)	
		#iterations	total time (s)	total	in CHOLMOD
stretched sheet	260k	105	6564.25	62.52	47.47
sheared rectangle	20k	—	—	—	—
torus	40k	66	448.68	6.80	4.42
balloon	40k	204	1651.71	8.10	5.40
teabag	30k	229	1347.81	5.89	4.21
teddy	98k	140	2259.93	16.14	10.30
twisted cylinder	960k	596	10183.8	17.09	12.86

Table 4. Timing for an alternative *StVK* setup where a problem-specific “flat” state is used as the initial guess rather than the *TFT* base mesh (see Section F.4). As in Table 3, the *StVK* columns give the number of iterations, and wall clock time, when we terminate the simulation due to small gradient residual or maximum iterations. The last two columns list the average wall clock time required by one *StVK* iteration, and the amount of that time spent specifically inside CHOLMOD. The sheared rectangle experiment failed completely (the simulation exploded after a few iterations, due to the excessive strain in the initial guess). Notice that the *TFT* initial guess leads to faster static solvers in all cases.

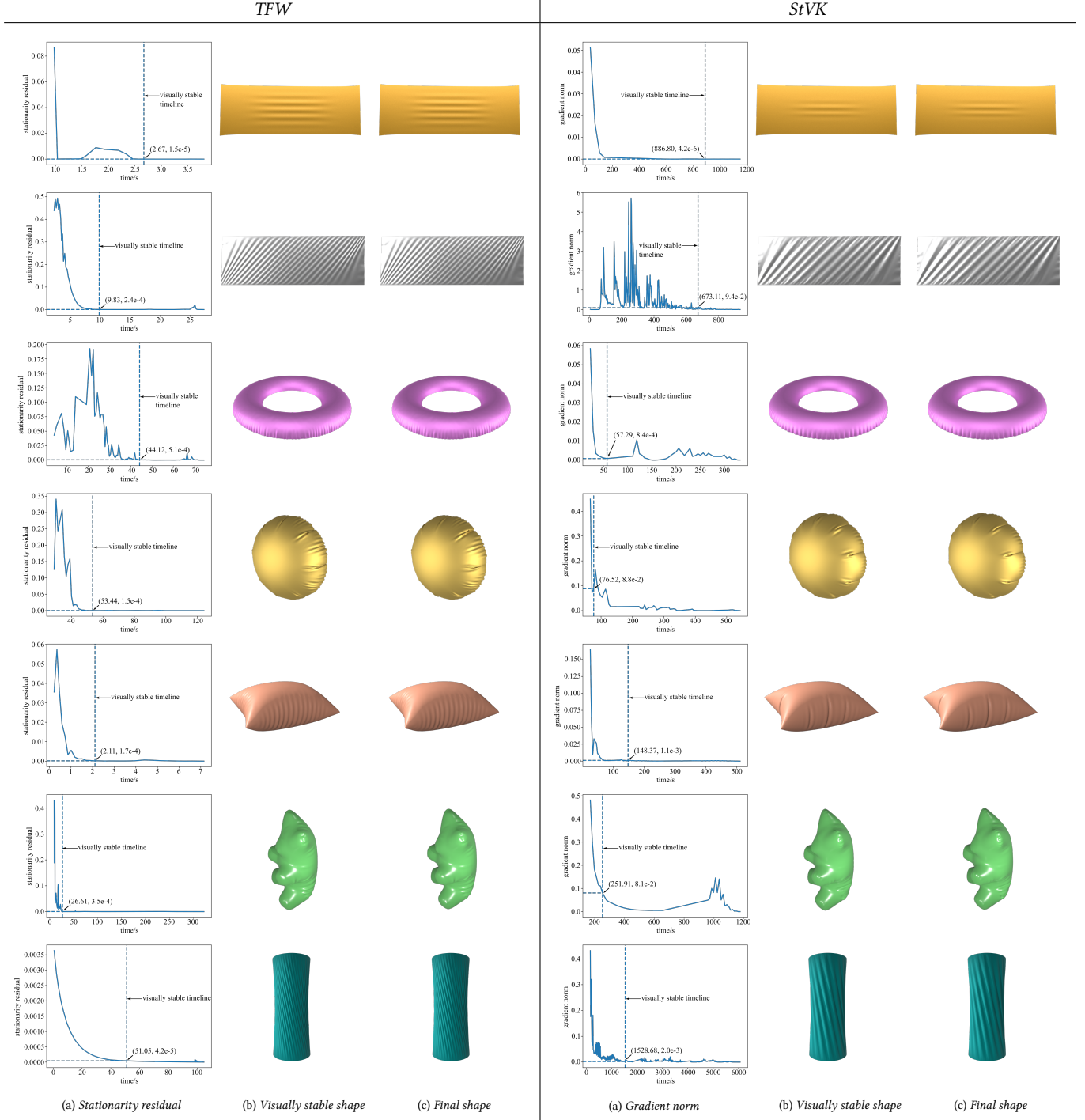


Fig. 24. Residual vs. wall clock time, plotted for each of our examples. For *StVK*, the residual is simply the gradient norm; for *TFW*, the residual includes projection onto the constraints in Equation 39, i.e.  $\|\nabla L\| = \|\nabla E^{\text{wf}} + S^T y + C^T z\|$ , where  $y, z$  are Lagrange multipliers. The dashed lines indicate the time where each simulation becomes visually stable, based on manual inspection. All norms are the infinity norm. In the second and third column, we show stills of the wrinkled surface at the visually stable time (*middle column*) and at the end of the experiment (*right column*), after we let the simulation run significantly past the visually stable time (we halt each simulation after 1000 iterations, or when the residual norm reaches  $10^{-6}$ , whichever comes first). For more information about each experiment, please refer to Table 1 in the main text.