Tensor-Train Approximation for High-Dimensional Economic Models*

Johannes Brumm Jakob Hußmann

December 30, 2025

Abstract

We introduce a scalable and broadly applicable method to compute accurate global solutions of high-dimensional dynamic stochastic models. Our approach leverages the tensor train format for function approximation, exploiting latent low-rank structures commonly present in economic models. While able to capture complex nonlinearities, the number of parameters of the tensor train approximation (TTA) grows only linearly in dimensions, significantly mitigating the curse of dimensionality. We show that TTA can be integrated naturally into standard iterative solution schemes, can capture irregularly shaped ergodic sets, is compatible with the endogenous grid method, makes quasi-analytical computation of expectations possible, and enables spectral solutions of continuous-time models, all in high dimensions. We illustrate the scalability of TTA by solving international real business cycle models of increasing dimension at only moderately increasing compute time without deteriorating accuracy. We demonstrate the versatility of TTA by solving heterogeneous agent models with large aggregate shocks. To approximate the wealth distribution efficiently, we introduce a moment-selection procedure that identifies the most informative moments from simulated data. In a model with wealth-tax shocks, the use of the first nine such moments as inputs to TTA reduces approximation errors by one order of magnitude relative to using only mean wealth.

Keywords: global solutions, tensor train decomposition, high-dimensional approximation, spectral methods, endogenous grid method, heterogeneous agents.
JEL Classification Codes: C61, C63, C68, E21, D31, D52.

*Contact details: Johannes Brumm, Karlsruhe Institute of Technology, johannes.brumm@kit.edu. Jakob Hußmann, Karlsruhe Institute of Technology, jakob.hussmann@kit.edu. We thank Simon Scheidegger, Yu-Cheng Yang, and participants at various seminars and conferences for helpful comments. We acknowledge financial support from the ERC project SOLG for Policy (101042908).

1 Introduction

Macroeconomic research has, over recent decades, increasingly focused on the role of heterogeneity, which has dramatically increased the complexity of solving the respective models. As a result, most studies have relied on perturbation techniques for aggregate dynamics, which, however, have limited accuracy beyond the neighborhood of the expansion point.[1] Addressing this issue calls for a global solution approach, which, however, is challenged by the curse of dimensionality. Recent advances aimed at overcoming that challenge fall mainly into two categories, sparse grids[2] and neural nets.[3] Yet, no single dominant technique has emerged due to inherent trade-offs. Sparse grid approaches, while robust and effective in medium dimensions, require too many points in high dimensions and lack flexibility in efficiently fitting ergodic sets. Neural networks, in contrast, are highly flexible and scalable, yet often exhibit overfitting, noisy solutions, and fragile convergence behavior.

This paper introduces a novel approach to solving high-dimensional economic models. It proposes tensor train approximation (TTA), which makes the use of tensor product bases in high dimensions possible, despite the exponential growth in the number of basis coefficients. It does so by approximating the d-dimensional tensor (or array) of basis coefficients by d separate 3-dimensional tensors, thereby reducing parameter growth from exponential to linear. TTA exploits latent low-rank structures in the basis coefficients of the tensor product basis and can efficiently and robustly be implemented with an alternating least squares (ALS) scheme. With this method at hand we can enjoy the upsides of tensor-product based approximations even in high dimensions. First, least-squares approximation of basis functions is grid-free and highly flexible, thus allowing for arbitrary ergodic sets and for applying the endogenous grid method (EGM) in high dimensions. Second, TTA features analytic integration and differentiation, which enables fast and exact computation of expectations and allows for spectral solutions of continuous-time models.

Compared to sparse grids, TTA offers much greater flexibility as it is not restricted to regular grids. Compared to neural networks, it runs much lower risk of overfitting and fragile convergence behavior due to less expressivity. Compared to both approaches, TTA allows for a more efficient approximation of economic policy functions, which often exhibit low curvature regions, smoothness, symmetry, and limited cross-

[1] For discrete-time linearizations, see Reiter (2009), Auclert et al. (2021), and Bayer and Luetticke (2020). In continuous time, based on Achdou et al. (2021), Ahn et al. (2018) provide linearizations of the Kolmogorov Forward Equation. A growing literature explores higher-order perturbations, including Bilal (2023) in continuous time, Bhandari et al. (2023), Bayer et al. (2025) in discrete time.

[2] Examples include Krueger and Kubler (2006) on sparse grids, Judd et al. (2014) on dimension-adaptive grids, Brumm and Scheidegger (2017) on locally adaptive grids, and Schaab and Zhang (2022) on adaptive sparse grids in continuous time.

[3] Notable contributions include Maliar et al. (2021), Azinovic et al. (2022), and Han et al. (2021).

dimensional interactions. As the TTA framework is agnostic to the choice of basis functions, this choice offers additional flexibility in adopting the algorithm to the characteristics of policy and value functions in specific applications.

To demonstrate the flexibility, efficiency, and scalability of TTA in solving dynamic economic models, we first apply it to the international real business cycle (IRBC) model as specified in Brumm and Scheidegger (2017). For the smooth version of the model we use a polynomial basis, which turns out to deliver high accuracy at modest polynomial order. For the non-smooth version, with occasionally binding irreversible-investment constraints, we employ a piecewise linear hierarchical basis. In both model versions the computational time increases only by two orders of magnitude between the 11- and 51-dimensional cases, while the average Euler error remains well below 0.001% (0.01%) in the smooth (non-smooth) case.[4] In addition to excellent scaling behavior, the IRBC model allows us to showcase two powerful features of TTA. First, quasi-analytical computation of expectations, which dramatically improves both computational performance and accuracy. Second, a high-dimensional endogenous grid method that speeds up the algorithm further.

To illustrate the broad applicability of TTA, we also apply it to heterogeneous agent models in continuous time. To ensure that the distribution — the primary source of multidimensionality in these models — meaningfully affects household decisions, we introduce large wealth tax shocks. We find that in the absence of such large aggregate shocks, the mean suffices to summarize the distribution. After introducing a stochastic wealth tax of 20% occurring with a 2.5% probability per quarter, however, additional moments are required to maintain sufficient accuracy. Incorporating additional distributional moments, selected using a novel a posteriori model reduction procedure, based on dynamic mode decomposition, substantially mitigates the master equation errors. Going from the three dimensional specification (with one distributional moment) to the eleven dimensional specification (with nine moments), reduces the average error by one order of magnitude. The heterogeneous agent application demonstrates two key insights. First, TTA can effectively handle the workhorse models of modern macroeconomics, even in the presence of large aggregate shocks. Second, TTA enables spectral solutions of continuous-time models.

All in all, the different applications presented in this paper demonstrate that TTA is a scalable, flexible, and broadly applicable method to compute accurate global solutions of high-dimensional dynamic stochastic models. We believe that TTA adds a great new tool to macroeconomists' toolbox — a method that might turn out to be the

[4]By comparison, it is very hard to bring errors down to such levels with (adaptive) sparse grids, see Figures 8 and 11 in Brumm and Scheidegger (2017). Moreover, sparse grids require (even when only a level three grid is employed) an increase in computation time of more than four orders of magnitude for such an increase in dimensionality, see Figure 15 in Brumm and Scheidegger (2017).

most efficient and practical choice for many applications that require global approximation yet feature latent low-rank structures that TTA can economize on much better than other global methods.

Related Literature. This paper connects to two strands of the literature. The first is on global solution techniques for high-dimensional economic models, including heterogeneous agent models; the second is on tensor decomposition methods in general, and on the tensor train decomposition in particular.

Recent advances in global solution methods for dynamic stochastic economic models can broadly be categorized into grid-based methods and machine learning approaches. A prominent grid-based approach is the use of sparse grids, which significantly mitigate the curse of dimensionality by reducing the exponential growth of grid sizes. Krueger and Kubler (2006) introduce sparse grids with polynomial basis functions to economics. Judd et al. (2014) extend this framework by incorporating anisotropic grids and adaptive domain selection. Brumm and Scheidegger (2017) further advance the method by employing hierarchical basis functions, enabling local adaptivity and thereby adding another layer of sparsity. Schaab and Zhang (2022) apply adaptive sparse grids to continuous time models. Brumm et al. (2022) provide an overview of the use of (adaptive) sparse grids in economics.

A growing literature applies machine learning techniques in general and (deep) neural networks in particular to approximate value and policy functions in dynamic models. Maliar et al. (2021) and Azinovic et al. (2022) demonstrate that deep neural networks can be trained to solve Bellman equations and first-order equilibrium conditions, even in models with high-dimensional state spaces. These techniques have enabled a range of applications, including household finance (Gorodnichenko et al., 2021), monetary policy (Nuño et al., 2024), climate change (Folini et al., 2025), asset pricing (Azinovic and Zemlicka, 2024), and structural estimation (Kase et al., 2022). While much of this work focuses on discrete-time models, other studies consider models in continuous time. For instance, neural networks have been applied to continuous-time economic models models (e.g., Fernández-Villaverde et al., 2020; Gopalakrishna, 2021; Sauzet, 2021), finance models (Duarte et al., 2024), and to problems formulated as backward stochastic differential equations (Huang, 2023). A related set of contributions applies machine learning techniques to heterogeneous agent models. Fernández-Villaverde et al. (2023) use neural networks to approximate the perceived law of motion in rational expectations models. Kahou et al. (2021) exploit symmetries in the distributional moments to simplify high-dimensional expectations. Han et al. (2021) propose a deep learning framework to approximate value functions while jointly choosing optimal distributional statistics. Payne et al. (2025) adapt similar tools to search-and-matching environments. In continuous time, Gu et al. (2024) solve the master equation

in heterogeneous agent models using deep learning. Fernández-Villaverde et al. (2024) provides a comprehensive overview of this emerging field.

In models with a continuum of heterogeneous agents, a key challenge is to compress the infinite-dimensional asset distribution without losing too much relevant information. Approaches range from relying just on aggregate capital as in Krusell and Smith (1998) to global polynomial representations as in Schaab (2020) and even full histogram discretizations as in Gu et al. (2024). To select informative state variables, Ahn et al. (2018) apply an a priori model reduction technique, while Han et al. (2021) jointly learn optimal generalized moments and solve the model equations. We propose an a posteriori simulation-based model reduction approach relying on dynamic mode decomposition (DMD).

The second strand of the literature to which this paper contributes concerns tensor train (TT) methods for high-dimensional problems. Following the seminal work of Oseledets (2011), the TT decomposition has gained significant attention as a tool for representing high-dimensional tensors with linear scaling in dimensionality. Holtz et al. (2012) propose an efficient and numerically stable alternating least squares (ALS) algorithm for optimization problems in the TT-format. Grasedyck and Krämer (2019) study optimal rank selection for tensor decompositions, offering complementary strategies to enhance the convergence and stability of ALS-type methods. Gorodetsky et al. (2019) develop a continuous analogue of the TT decomposition, enabling the representation of multivariate functions rather than discrete arrays. Bigoni et al. (2016) introduce a spectral tensor train decomposition that combines the TT structure with spectral polynomial approximation, further improving accuracy in function approximation. Recent work has applied TT techniques to the solution of high-dimensional PDEs and control problems. Dektor et al. (2021) develop a rank-adaptive method that combines functional TT expansions with a dynamic algorithm that adjusts ranks during time integration. Dolgov et al. (2021) apply TT methods to solve high-dimensional Hamilton-Jacobi-Bellman equations, demonstrating their potential in optimal control problems. Richter et al. (2021, 2024) extend TT-based solvers to parabolic PDEs and compare their performance favorably to neural network-based approaches, emphasizing robustness and interpretability. Bachmayr (2023) offer a comprehensive review of low-rank tensor techniques, including TT formats, covering theoretical underpinnings, numerical algorithms, and applications to PDEs and high-dimensional optimization.

2 Tensor Train Approximation

This section explains how to approximate multi-dimensional functions via TTA. First, we show how to write the coefficients of a tensor-product basis for approximating d-

dimensional functions as an order d tensor. Second, we demonstrate how to alleviate the curse of dimensionality by approximating the latter with a so-called tensor train composed of d different order three tensors. Third, we provide an ALS algorithm for computing the coefficients of such a TTA efficiently and robustly. Finally, we point out that derivatives and integrals of TTAs can be computed analytically. In Appendix A we show that the TTA algorithm is a special case of a broader class of quadratic problems that can be minimized efficiently by ALS. We construct another special case of that class in Section 4 when solving master equations.

2.1 Tensor Representation of Basis Coefficients

To approximate a multi-dimensional function, $f : \mathbb{R}^d \to \mathbb{R}$, $d > 1$, the most straightforward approach is to represent it as a product of one-dimensional basis functions. Assuming, for ease of exposition, that all dimensions are treated equally, we allow for m one-dimensional basis functions, $\phi_1, \ldots, \phi_m$, where $\phi_i : \mathbb{R} \to \mathbb{R}$. Thus, to approximate f at $x \in \mathbb{R}^d$, denoted by $\mathcal{F}(x)$, we evaluate $m \cdot d$ basis functions $\phi_{i_1}, \ldots \phi_{i_d}$ with $i_k = 1, \ldots, m$ and $k = 1, \ldots, d$.[5] To map these basis functions to the corresponding scalar $\mathcal{F}(x)$, we define an order d tensor A containing all basis coefficients. An order d tensor is a multidimensional generalization of a matrix. The entry $A[i_1 \ldots i_d]$ indicates the weight by which the product of the respective basis functions contributes to the value of $\mathcal{F}(x)$:

$$\mathcal{F}(x) = \sum_{i_1=1}^{m} \ldots \sum_{i_d=1}^{m} A[i_1, \ldots, i_d] \cdot \phi_{i_1}(x_1) \cdot \ldots \cdot \phi_{i_d}(x_d). \tag{1}$$

As A contains weights for all permutations of bases $\phi_1, \ldots, \phi_m$ across d dimensions it consists of m^d entries and is thus clearly susceptible to the curse of dimensionality. Computing all its entries becomes rapidly infeasible as d increases. However, doing so might not be necessary for achieving an accurate approximation. Instead of using A in its entirety, we can employ a lower-dimensional object to approximate it. This is exactly what TTA does, as we explain next. Of course, such an approach is only promising if A possesses a latent low-rank structure. In economic applications this is often the case, as the applications in Sections 3 and 4.

2.2 Tensor Train Format

Following Oseledets (2011), we approximately factorize an order-d tensor A as the product of d order-three tensors W^i, each of size $r_{i-1} \times m \times r_i$, with $r_i \geq 1$, $r_0 = r_d = 1$.

[5] In the applications below, we use polynomial basis functions and piecewise linear hierarchical basis functions.

We call $\mathcal{T} = (W^1, \ldots, W^d)$ the tensor train decomposition of A and W^i a tensor core or wagon of the tensor train. The ranks $r_0, \ldots, r_d$ govern the size of the wagons W^i, and control the dependencies across wagons. TTA approximates each entry of A through a chain of contractions over the intermediate indices linking consecutive cores. To define how exactly it does so, we employ the so-called Einstein convention, summing over repeated indices and letting the free indices determine the output tensor's shape:[6]

$$A \approx \left(W^1_{i_1 j_1} \cdot W^2_{j_1 i_2 j_2} \cdot \ldots \cdot W^{d-1}_{j_{d-2} i_{d-1} j_{d-1}} \cdot W^d_{j_{d-1} i_d} \right)_{i_1 \ldots i_d}. \tag{2}$$

Suppose $d = 3$, $m = 3$, $r = 2$, then we are contracting a 3×2 matrix with a $2 \times 3 \times 2$ tensor to obtain a $3 \times 3 \times 2$ tensor, which we then contract with a 2×3 tensor to obtain a $3 \times 3 \times 3$ tensor. In this small example, the tensor train has 24 free parameters, just slightly less than the 27 of A. Yet with $d = 10$, $m = 3$, $r = 2$, the tensor train format already reduces the degrees of freedom from 59.049 to 108. In general, when all basis sizes equal m as assumed above[7] and the ranks satisfy $r_\ell \leq r$, the number of free parameters of the TTA $\mathcal{T}$ is bounded by dmr^2 as compared to m^d for A. Thus, the TTA approach alleviates the curse of dimensionality: with fixed m and r, the number of parameters grows linearly instead of exponentially in d. Given a tensor train $\mathcal{T}$, we can express an approximator of f as follows:

$$\mathcal{F}(x; \mathcal{T}) = \left(W^1_{i_1 j_1} \cdot \phi(x_1)_{i_1} \right)_{j_1} \cdot \left(W^2_{j_1 i_2 j_2} \cdot \phi(x_2)_{i_2} \right)_{j_1 j_2} \cdot \ldots \cdot \left(W^d_{j_{d-1} i_d} \cdot \phi(x_d)_{i_d} \right)_{j_{d-1}}. \tag{3}$$

Thus, in each dimension ℓ we evaluate the basis functions, obtaining the vector $\phi(x_\ell) = (\phi_1(x_\ell), \ldots, \phi_m(x_\ell))$, which we then contract with the core W^ℓ resulting in an order-two tensor, except in case of $\ell = 1$ or $\ell = d$ where we get an order-one tensor otherwise. These d tensors are then contracted successively with each other to finally obtain the scalar $\mathcal{F}(x; \mathcal{T})$. Evaluations of the approximator are of computational complexity $\mathcal{O}(mdr^2)$; thus, function evaluations scale efficiently in dimensions d. To convey the intuition for TTA, we introduce diagrammatic notation borrowed from physics. Nodes represent cores and edges illustrate summations. Figure 1 provides a diagrammatic illustration of the decomposition of A and the tensor train approximator $\mathcal{F}(x; \mathcal{T})$.

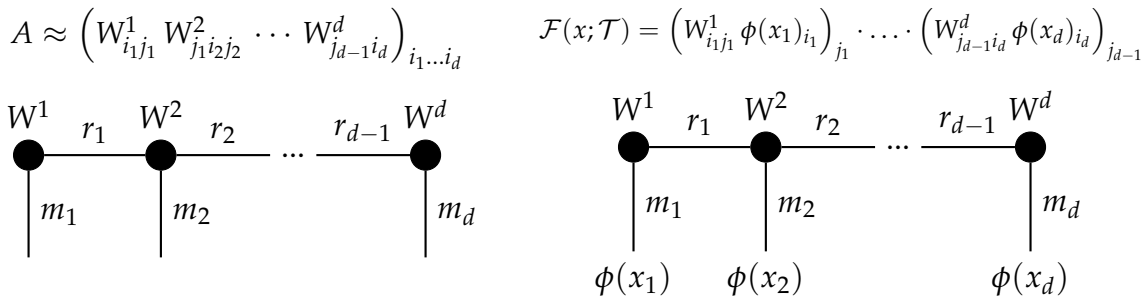
[6]We use the notation as follows: For all tensors involved in contraction we explicitly denote their dimension indices. If an index present in the original tensors is missing in the result, this implies summation over that index. For example the contraction of two consecutive cores is written as

$$U[j_1, i_2, i_3, j_3] = \left(W^2_{j_1 i_2 j_2} \cdot W^3_{j_2 i_3 j_3} \right)_{j_1 i_2 i_3 j_3} [j_1, i_2, i_3, j_3] = \sum_{j_2=1}^{r_2} W^2_{j_1 i_2 j_2}[j_1 i_2 j_2] \cdot W^3_{j_2 i_3 j_3}[j_2 i_3 j_3].$$

Indices in brackets signal specific entries of the tensor, while indices in lower-subscripts denote all entries over this dimension of the tensor.

[7]Note that basis sizes m need not be uniform across cores for the method to work. In some of the applications below, basis size will vary across dimensions.

Figure 1: Tensor Train Formulas and Diagrammatic Notation



(a) TT-Decomposition of A

(b) TT-Approximator $\mathcal{F}(x;\mathcal{T})$

Diagrammatic tensor train notation. Filled nodes $W^1, \ldots, W^d$ are TT wagons; horizontal lines carry the TT ranks r_k; downward lines are indices of size m_k. Left: the coefficient tensor A represented as a tensor train. Right: $\mathcal{F}(x;\mathcal{T})$ obtained from the same chain by contracting with the local basis vector $\phi(x_k)$. The expressions above each panel spell out the contractions in index form.

2.3 Optimizing Tensor Trains — The TTA Algorithm

We now present an algorithm for TT-based function approximation. It computes a TT $\mathcal{T}$ such that $\mathcal{F}(x;\mathcal{T})$ fits the sample points $\{(x^n, y^n)\}_{n=1}^N \subset \mathbb{R}^d \times \mathbb{R}$ in a least squares sense. We first formulate the associated least-squares problem and then introduce the Alternating Least Squares (ALS) scheme for solving it. We denote the resulting TTA with basis sizes m and ranks r by $\mathcal{T} = \mathcal{A}\left(m, r, \{x^n, y^n\}_{n=1}^N\right)$.

Optimization. For given basis shape m and rank r, we fit the TTA approximator $\mathcal{F}(x;\mathcal{T})$ by choosing a tensor train $\mathcal{T} = (W^1, \ldots, W^d)$ to minimize the regularized least squares error on the sample set $\{(x^n, y^n)\}_{n=1}^N$:

$$\min_{W_1,\ldots,W_d} \frac{1}{N} \sum_{i=1}^{N} ||y^i - \mathcal{F}(x^i; W^1, \ldots, W^d)||_2^2 + \lambda \sum_{k=1}^{d} ||W^k||_F^2, \tag{4}$$

where λ is the regularization parameter, and $||\cdot||_F$ is the Frobenius norm. In the above problem the coefficients of different cores interact multiplicatively making the global problem non-convex. Following Holtz et al. (2012), we use an alternating least-squares (ALS) scheme to circumvent this problem. Fixing all cores except one, W^k, makes the mapping from W^k to $\mathcal{F}(x^i; W^1, \ldots, W^k, \ldots W^d)$ linear, so finding the optimal W^k reduces to a simple least-squares problem:

$$\min_{W_k} \frac{1}{N} \sum_{i=1}^{N} ||y^i - \mathcal{F}(x^i; W_1, \ldots, W_k, \ldots W_d)||_2^2 + \lambda ||W^k||_F^2. \tag{5}$$

Exploiting this insight, ALS sweeps over the cores, left to right, and back, solving the local subproblems until convergence. In what follows, we discuss how the single-core

subproblem (5) can be written in vectorized form and how the cores can be orthogonalized for numerical stability. Algorithm 1 presents the complete TTA algorithm, while Figure 2 sketches the algorithm in diagrammatic notation.

Vectorization. The single-core subproblem (5) can be written as

$$\min_{v^k} \quad \frac{1}{N}||y - B^k v^k||_2^2 + \lambda||v^k||_2^2, \tag{6}$$

where $v^k = \text{vec}\left(W^k\right)$ represents the wagon to be optimized, and B^k contains all information about other wagons and about basis function evaluations at all sample points. B^k is constructed as follows: First, let $\mathcal{X}^k_{i_k n} = \phi_{i_k}\left(x^n_k\right)$ denote the basis values in dimension k for sample point n. Then define the left and right stacks, $\mathcal{L}^k$ and $\mathcal{R}^k$, starting with $\mathcal{L}^0 = \mathcal{R}^d = \vec{1} \in \mathbb{R}^N$ and recursively proceeding with

$$\mathcal{L}^k = \left(\mathcal{L}^{k-1}_{j_{k-2}n} \cdot W^{k-1}_{j_{k-2}i_{k-1}j_{k-1}} \cdot \mathcal{X}^{k-1}_{i_{k-1}n}\right)_{j_{k-1}n}, \quad \mathcal{R}^k = \left(W^{k+1}_{j_k i_{k+1} j_{k+1}} \cdot \mathcal{X}^{k+1}_{i_{k+1}n} \cdot \mathcal{R}^{k+1}_{j_{k+1}n}\right)_{j_k n}. \tag{7}$$

Finally, define

$$\mathcal{B}^k = \left(\mathcal{L}^k_{j_{k-1}n} \cdot \mathcal{X}^k_{i_k n} \cdot \mathcal{R}^k_{j_k n}\right)_{n j_{k-1} i_k j_k}, \tag{8}$$

and reshape it into a matrix $B^k \in \mathbb{R}^{N \times (r_{k-1} m_k r_k)}$.

Orthogonalization. To maintain numerical stability, Holtz et al. (2012) propose to keep a left-orthonormal prefix of cores and a right-orthonormal suffix during the ALS sweeps. For the k-th core W^k define its left and right unfoldings by grouping indices

$$\text{unf}_L(W^k) \in \mathbb{R}^{(r_{k-1}m_k)\times r_k}, \quad \text{unf}_R(W^k) \in \mathbb{R}^{(r_k m_k)\times r_{k-1}}. \tag{9}$$

We call W^k left-orthonormal and right-orthonormal, respectively, if it satisfies:

$$\text{unf}_L(W^k)^\intercal \text{unf}_L(W^k) = I, \quad \text{unf}_R(W^k)^\intercal \text{unf}_R(W^k) = I. \tag{10}$$

During a left-to-right sweep we enforce left-orthonormality via a (thin) QR decomposition,

$$\text{unf}_L(W^k) = QR, \quad Q^\intercal Q = I, \tag{11}$$

denoting the outcome by $[Q, R] = \text{qrd}(\text{unf}_L(W^k))$. We then reshape Q, $\text{res}(Q) \in \mathbb{R}^{r_{k-1}\times m_k \times r_k}$, and set $W^k := \text{res}(Q)$. The triangular factor, R, is absorbed into the next core, $W^{k+1} := \text{res}\left(\text{unf}_R\left(W^{k+1}\right) \cdot R^\top\right)$, so that the overall tensor remains unchanged but the next subproblem is better conditioned. In the right-to-left sweep

right-orthonormality is analogously enforced. Details of the procedure are given in Algorithm 1 and displayed in Figure 2.

Algorithm 1 Tensor Train Approximation (TTA)

Require: Basis dimensions $m = (m_1, \ldots, m_d)$; ranks $r = (r_0, \ldots, r_d)$ with $r_0 = r_d = 1$; data points $\{x^n, y^n\}_{n=1}^N$ with $x^n \in \mathbb{R}^d$, $y^n \in \mathbb{R}$; initial cores $W^1, \ldots, W^d$ with $W^k \in \mathbb{R}^{r_{k-1} \times m_k \times r_k}$; regularization λ.

Alternating Least Squares

Build initial stacks $\mathcal{L}^k, \mathcal{R}^k$ using equation (7).

Set $\tilde{\xi} = \infty$ and $\epsilon = \infty$.

while $\epsilon > \varepsilon$ **do** ▷ Macro-Iterations (sweeps)

for $k = 1, \ldots, d-1$ **do** ▷ First Half-Sweep

Compute B^k using equation (8).

Solve $v^k := \arg\min_v ||B^k v - y||_2^2 + \lambda ||v||_2^2$.

Compute $[Q, R] = \mathrm{qrd}(\mathrm{unf}_L(\mathrm{res}(v^k)))$.

Set $W^k := \mathrm{res}(Q)$, $W^{k+1} := \mathrm{res}\left(\mathrm{unf}_R\left(W^{k+1}\right) \cdot R^\top\right)$.

Update left stack $\mathcal{L}^{k+1}$ using equation (7).

end for

for $k = d, \ldots, 2$ **do** ▷ Second Half-Sweep

Compute B^k using equation (8).

Solve $v^k := \arg\min_v ||B^k v - y||_2^2 + \lambda ||v||_2^2$.

Compute $[Q, R] = \mathrm{qrd}(\mathrm{unf}_R(\mathrm{res}(v^k)))$.

Set $W^k := \mathrm{res}(Q)$, $W^{k-1} := \mathrm{res}\left(\mathrm{unf}_L\left(W^{k-1}\right) \cdot R^\top\right)$.

Update right stack $\mathcal{R}^{k-1}$ using equation (7).

end for

Compute $\xi = ||y - \tilde{y}||_2$, set $\epsilon := \tilde{\xi} - \xi$, and $\tilde{\xi} := \xi$.

end while

Return Tensor train approximation $\mathcal{T} = \mathcal{A}\left(m, r, \{x^n, y^n\}_{n=1}^N\right) = \left(W^1, \ldots, W^d\right)$.

2.4 Analytic Differentiation and Integration

As it will become extremely useful below, we now point out that function representations in TT-format admit efficient rules for differentiation and integration. Let $\mathcal{F}(x; \mathcal{T})$ be a TT approximator, and let $\mathcal{L}^k_{j_{k-1}}$ and $\mathcal{R}^k_{j_k}$ be the left and right stacks obtained by contracting all cores except the k-th with the local basis evaluations as in equation (7).

Figure 2: Visualization of an ALS Iteration

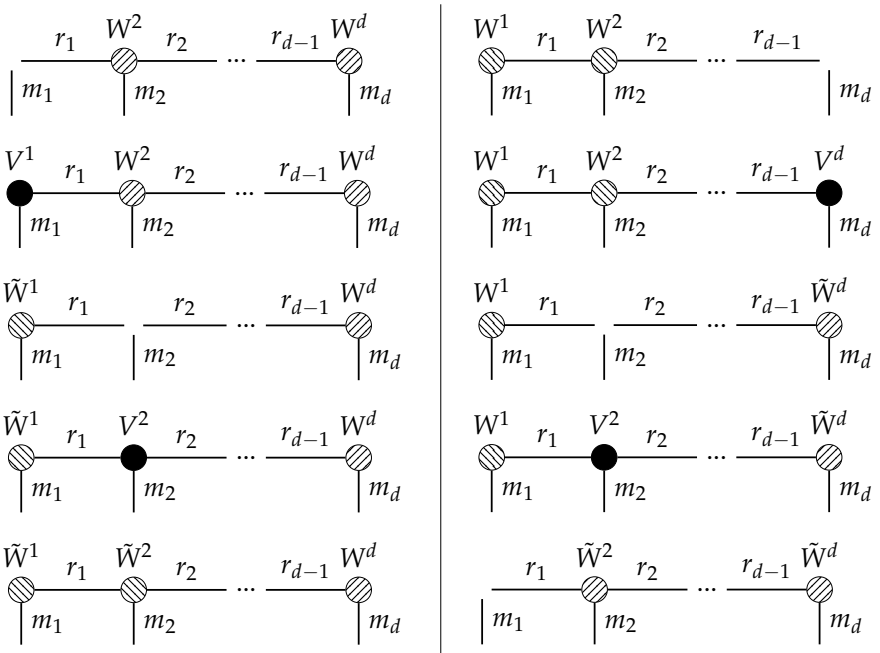


Illustration of one iteration of alternating least squares (ALS). A forward (left–to–right) half-sweep depicted on the left followed by a backward (right–to–left) half-sweep depicted on the right. Hollow nodes indicate the yet-unknown subproblem solution at the current position; filled nodes indicate the just-solved subproblem; an upward (downward) sloping texture marks a right- (left-) orthonormal core.

We can then take the derivative of $\mathcal{F}(x;\mathcal{T})$ with respect to x_k by replacing the basis vector in dimension k with its derivative, and contract the cores in the standard way:

$$\frac{\partial^n \mathcal{F}(x;\mathcal{T})}{\partial x_k^n} = \mathcal{L}^k_{j_{k-1}} \cdot W^k_{j_{k-1} i_k j_k} \cdot \left(\partial^n_{x_k} \phi(x_k)\right)_{i_k} \cdot \mathcal{R}^k_{j_k}. \tag{12}$$

To calculate the indefinite integral of $\mathcal{F}(x;\mathcal{T})$ with respect to x_k one simply replaces the basis vector in dimension k with its anti-derivative:

$$\int \mathcal{F}(x;\mathcal{T})\,\mathrm{d}x_k = \mathcal{L}^k_{j_{k-1}} \cdot W^k_{j_{k-1} i_k j_k} \cdot \left(\int \phi(x_k)\,\mathrm{d}x_k\right)_{i_k} \cdot \mathcal{R}^k_{j_k} \;+\; C(x_{-k}), \tag{13}$$

where $C(x_{-k})$ is the integration constant that may depend on all variables except x_k. For definite integrals, we get:

$$\int_{a_k}^{b_k} \mathcal{F}(x;\mathcal{T})\,\mathrm{d}x_k = \mathcal{L}^k_{j_{k-1}} \cdot W^k_{j_{k-1} i_k j_k} \cdot \left(\Phi(b_k) - \Phi(a_k)\right)_{i_k} \cdot \mathcal{R}^k_{j_k}, \quad \Phi' = \phi. \tag{14}$$

The formulas in (12)–(14) reveal that partial derivatives and one–dimensional integrals have the same asymptotic cost as function evaluations, since $\phi(x_k)$ is merely replaced by $\partial^n_{x_k}\phi(x_k)$ or $\Phi(x_k)$. This fact will prove extremely useful in two respects: First, it will

allow us, in Section 3, to compute expectations in high-dimensional models both accurately and efficiently. Second, it will allow us, in Section 4, to solve partial differential equations as apparent in continuous-time heterogeneous-agent models.

3 Solving High-Dimensional IRBC Models with TTA

To demonstrate the accuracy and scalability of the TTA approach, we now apply it to the international real business cycle (IRBC) model, as presented in Brumm and Scheidegger (2017). After briefly describing the model, we explain how TTA allows for quasi-analytical computation of high-dimensional expectations. We then show how TTA can naturally be embedded in the EGM alorithm and report results on the accuracy and scalability of the resulting algorithm as applied to both the smooth IRBC model and a non-smooth version of it.

3.1 IRBC Model

Physical Economy. The model features M countries, indexed by $j = 1, \ldots, M$, each utilizing its capital stock to produce output via Cobb-Douglas production with fixed labor supply. The output good can be allocated to either consumption, c_j, or investment, χ_j. Consumption generates utility through an additively separable utility function with discount factor β and per-period utility function of the CRRA type with risk aversion γ_j, varying across countries. Investment is subject to adjustment costs $k_j \cdot \phi \cdot g_j^2/2$, where $g_j \equiv k_j'/k_j - 1$ and $\phi > 0$. Additionally, capital depreciates at a rate $\delta > 0$. Countries productivity is given by

$$\ln a_j' = \rho \ln a_j + \sigma(e_j' + z'), \tag{15}$$

where the country specific shocks, e_j', and the global shock, z', are assumed to be i.i.d. standard normal shocks that are independent both across time and from one another. We parameterize the model as Brumm and Scheidegger (2017), with the exception of a lower depreciation rate that gives us more frequent exposure to the irreversibility constraint in the simulations. All parameters are reported in Table 1.

Complete Markets. Following Kollmann et al. (2011) and Brumm and Scheidegger (2017), we assume complete markets. This assumption implies that the decentralized competitive equilibrium allocation can be characterized as the solution to a social planner's problem. Subject to aggregate resource constraints, the planner maximizes the weighted sum of country-specific utilities, where each country's utility is weighted by τ_j, a welfare weight that depends on its initial capital stock.

Table 1: Parameterization of IRBC model

Parameter	Symbol	Value
Discount Factor	β	0.99
EIS of country j	γ_j	$[0.35, 1]$
Capital share	ζ	0.36
Depreciation	δ	0.005
Std. of log-productivity	σ	0.01
Autocorrelation of log-productivity	ρ	0.95
Intensity of capital adjustment costs	ϕ	0.5
Aggregate productivity	A	$(1-\beta(1-\delta))/(\zeta\beta)$
Welfare weight of country j	τ_j	A^{1/γ_j}

Recursive Equilibrium Conditions. The equilibrium conditions in each period consist of the optimality conditions for investment in each country's capital,

$$\lambda\left[1+\phi g_j\right] = \beta\mathbb{E}_t\left\{\lambda'\left[a_j' A\zeta(k_j')^{\zeta-1} + 1 - \delta + \frac{\phi}{2}g_j'(g_j'+2)\right]\right\}, \quad j=1,\ldots,M, \tag{16}$$

and the aggregate resource constraint, given by

$$\sum_{j=1}^{M}\left(a_j A(k_j)^{\zeta} + k_j\left(1-\delta-\frac{\phi}{2}g_j^2\right) - k_j' - \left(\frac{\lambda}{\tau_j}\right)^{-\gamma_j}\right) = 0. \tag{17}$$

These equations jointly determine the capital choice of each country j, k_j', and the Lagrange multiplier on the resource constraint, λ. For the derivation of these conditions, we refer the reader to Brumm and Scheidegger (2017).

3.2 Quasi-Analytic Expectations with TTA

When solving high-dimensional stochastic models, not only function approximation but also the computation of expectations poses a formidable challenge. In case of the IRBC model, the $M+1$ dimensional integral in equation (16) has to be computed. Previous papers solve the high-dimensional IRBC model either by resorting to compute-intense Monte-Carlo integration or monomial rules, which exhibit polynomial growth in the number of dimensions (see Kollmann et al., 2011). Quadratically growing monomial rules achieve decent accuracy at the expense of becoming too compute intensive at medium scale already, while monomial rules that grow only linearly lack accuracy.[8]

[8]Employing such a monomial rule in both the time iteration and the error evaluation step, Brumm and Scheidegger (2017) deliberately sidestep the challenge of computing accurate high-dimensional expectations to focus on high-dimensional function approximation.

Due to the stark trade-off between speed and accuracy in high-dimensional numerical integration, an approach that allowed for calculating expectations analytically would be very welcome. Such an approach would not only be computationally efficient, it would also achieve greater accuracy by integrating over the entire distribution rather than relying on a finite number of evaluation points. TTA allows for such an approach, as we now show.

Integration across the dimensions of a tensor train can be performed analytically, as detailed in Section 2, by substituting the basis functions of the relevant dimension with their anti-derivative. However, this property alone does not fully resolve the expectations problem, since expectations are not simply integrals over policy functions — as can be seen in equation (16). To address this complication, we proceed in three steps. First, instead of using the natural recursive state of the economy, capital stocks, k_j, and productivities, a_j, of all countries, we use the following state:

$$x = (k_1, \ldots, k_M, \tilde{a}_1, \ldots, \tilde{a}_M, z) \in X \subset \mathbb{R}^{2M+1}, \tag{18}$$

where $\tilde{a}_j = \ln a_j - \sigma z$ is the log-productivity of country j before the impact of the global shock, z, is taken into account. Second, we define the term inside the expectations operator in country j's Euler equation, which we refer to as the expectations term,

$$\eta_j(x') = \lambda' \left[a'_j A \zeta (k'_j)^{\zeta - 1} + 1 - \delta + \frac{\phi}{2} g'_j (g'_j + 2) \right], \tag{19}$$

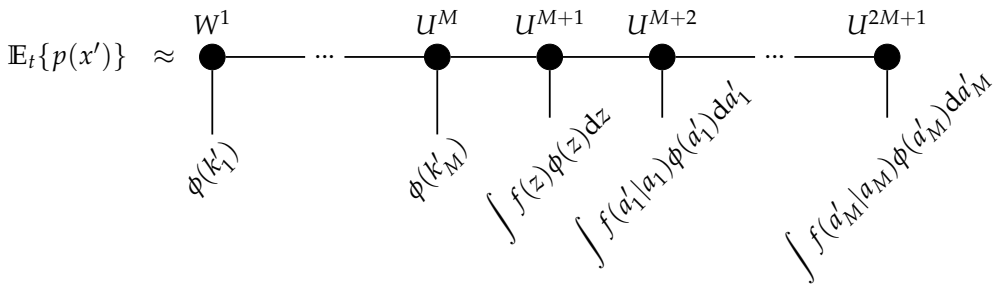
so that expectations can be computed as

$$\begin{aligned} \mathbb{E}_t\{\eta_j(x')\} = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} & f(\tilde{a}'_1 | \rho \ln a_1, \sigma) \cdot \ldots \cdot f(\tilde{a}'_M | \rho \ln a_M, \sigma) \cdot f(z' | 0, 1) \cdot \\ & \eta_j(k'_1, \ldots, k'_M, \tilde{a}'_1, \ldots, \tilde{a}'_M, z') \, d(z', \tilde{a}'_1, \ldots, \tilde{a}'_M), \end{aligned} \tag{20}$$

where $f(\cdot | \mu, \varsigma)$ denotes the probability density function of a normal distribution with mean μ and standard deviation ς. In the third step, we perform a change of basis for the individual productivities $\tilde{a}'_1, \ldots, \tilde{a}'_M$ and the global shock z'. Specifically, we replace the polynomial basis functions $\phi(x)$ with a precomputed "expectations basis", given by the integral of the product of the density function and the polynomial basis functions, $\int f(x|\mu, \sigma)\phi(x)\mathrm{d}x$.[9] This transformation allows us to directly incorporate the probability distribution into the tensor train structure, streamlining the computation

[9] While the choice of the expectations basis for z, which is distributed normally with mean zero, is straightforward, more care is required for the expectations over $\tilde{a}'_j$. Since $\ln \tilde{a}'_j$ follows an AR(1) process, it is also normally distributed with mean $\rho \ln a_j$, which depends on the current state. In practice, we compute the expectations basis on a very fine grid for μ and interpolate it based on the specific value of a_j. Note that this interpolation is one-dimensional, as it only affects the basis functions, and thus does not introduce additional complications related to the curse of dimensionality.

Figure 3: Expectation Formation with Tensor Trains



The figure shows how the integrals in equation (20) can be streamlined using the tensor train decomposition. The integrals address only isolated dimensions of the tensor train, thus their evaluation can be done directly at the corresponding core, by replacing the basis.

of expectations. For better intuition, Figure 3 illustrates this process using the graphical notation introduced in Section 2. Note that this quasi-analytic integration approach is only possible when integrating over random variables that are (conditionally) independent, which is the reason why we choose the state x as specified in equation (18) above.

3.3 Endogenous Grids in High Dimensions with TTA

There are two standard routes to enforcing the optimality system (16) - (**??**). Brumm and Scheidegger (2017) apply a numerical root finder to the equilibrium conditions and iterate on the policy function (known as time iteration). With TTA, we can take a more efficient path: a variant of the endogenous grid method of Carroll (2006). In the IRBC model EGM replaces a joint $M+1$-dimensional numerical solve with a one-dimensional solve. Since this approach produces irregular (endogenous) grids it is not applicable for grid based methods; TTA, however, is agnostic to the underlying sample, which makes it the natural companion for EGM.

Applied to the IRBC model, EGM works as follows. Fix a candidate policy η. For each exogenous state (z_t, a_t) fix the choice k_{t+1}, and compute the right-hand side of the Euler equation (16). With η given, the expectation is obtained quasi-analytically, according to the previous section. Given a solution candidate for λ_t we can rearrange the Euler equation (16) to obtain the endogenous state k_t analytically. Plug-in the endogenous state in the aggregate resource constraint (17) and update the solution candidate λ_t until it solves (17).

The procedure can by easily extended to accommodate the irreversibility constraint of the non-smooth model.[10]

[10] Details can be found in Appendix B.

3.4 The TTA Algorithm for Solving IRBC Models

We now present an algorithm that employs TTA to compute a recursive equilibrium of the IRBC model. The recursive state of the economy $x \in X$ consists of the country specific capital stocks k_j, transformed productivities $\tilde{a}_j$, and the global shock z (see equation (18)). The policy we solve for consists of the expectation-terms η_j defined in equation (19):[11]

$$p : X \to \mathbb{R}^M, p(x) = (\eta_1(x), \ldots, \eta_M(x)) . \tag{21}$$

The mapping p determines only M policy components, while the equilibrium system involves $M + 1$ unknowns. Hence, given $p(x)$, simulation requires solving for λ to enforce the aggregate resource constraint. [12]

A recursive equilibrium consists of a policy function p and an ergodic set $\mathcal{E} \subset X$, such that (i) policies satisfy the equilibrium conditions on the ergodic set and (ii) simulating the policies generates the ergodic set. Corresponding to conditions (i) and (ii), the algorithm consists of two levels, with the inner level computing policies that approximately satisfy optimality conditions (16) and (17) on a set of sample states $\mathcal{S}$ using EGM. The outer level, once the inner level converged, simulates the model to update the set $\mathcal{S}$ to get closer to the ergodic set.[13] A detailed description of the algorithm can be found in Algorithm 2. To measure the accuracy of the solution, we follow the standard approach of computing unit-free (relative) Euler equation errors for each of the M countries, as well as an error for the aggregate resource constraint. These measures reflect how closely the numerical solution satisfies the model's equilibrium conditions. For each country j, we use the tensor train representations of the policy function η_j, and determine the Lagrange multiplier λ from the resource constraint to infer the capital choice k'_j from equation (19). The Euler equation error for country j is then defined as:

$$EE^j = \beta \mathbb{E}_t \left\{ \eta_j(x') \right\} \cdot \left[\lambda (1 + \phi \cdot g_j) \right]^{-1} - 1, \tag{22}$$

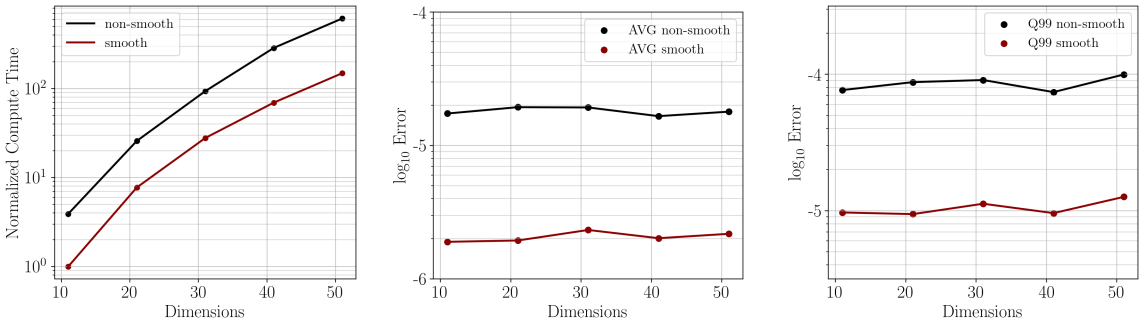
where expectations are computed using the same basis transformation employed during the model solution. This ensures consistency between the approximation and the

[11] The expectations terms are functions of the other policy choices. To employ the quasi-analytical expectations approach explained above, it is essential to directly approximate these terms via TTA. A practical advantage is that η_j is typically smoother than the underlying choice variables, which makes it easier to approximate accurately.

[12] While this may seem unusual, it is central to the efficiency of the approach: we approximate a comparatively simple and smooth object (latent low rank), and recover the remaining choice via a clearing condition.

[13] Of course, this approach is only possible as approximation via TTA can operate on irregular datasets. In high dimensions it becomes increasingly inefficient to approximate a model on a set of points by sampling from the hypercube enveloping the ergodic set, as the ratios of volumes between ergodic set and hypercube declines drastically. Moreover, the corners of the hypercube represent extreme circumstances with zero probability that unnecessarily require substantial non-linearity in the approximation.

Figure 4: IRBC Model Scaling and Errors



The left panel reports normalized computation time, normalized by the case of the 11-d smooth model. The horizontal axis spans model dimensions from 11 to 51, corresponding to 5 to 25 countries. Computation times are displayed on a base-10 logarithmic scale. The increase in computation time is modest: raising the dimensionality from 11 to 51 multiplies computation time by two orders of magnitude. The middle panel displays the average error in base-10 logarithmic units. The average error remains well below 10^{-5} and 10^{-4} in the smooth and the non-smooth model, respectively. The right panel displays the 99th percentile error, which is around one order of magnitude higher than the average error in both models. Accuracy remains approximately the same with increasing dimensions.

evaluation of expectations. For the aggregate resource constraint, we define the relative error as:

$$\frac{\sum_{j=1}^{M}\left[a_j A k_j^{\zeta}+k_j\left((1-\delta)-\frac{\phi}{2}g_j^2\right)\right]-k_j'-\left(\frac{\lambda}{\tau_j}\right)^{-\gamma_j}}{\sum_{j=1}^{M}\left[a_j A k_j^{\zeta}+k_j\left(-\frac{\phi}{2}g_j^2\right)\right]}. \tag{23}$$

These error measures are computed along a simulation path of 10,000 periods, using a sequence of shocks that is independent from the one used to construct the state set $\mathcal{S}$ in the solution algorithm. This allows us to assess out-of-sample accuracy. A key advantage of our approach is that the expectations in the Euler equation are computed quasi-analytically. In contrast, approaches based on numerical quadrature—such as the monomial integration method used by Brumm and Scheidegger (2017)—introduce an additional source of approximation error due to the quadrature itself. By contrast, our method eliminates this source of error entirely, as expectations are evaluated exactly (within the functional basis).

3.5 Accuracy and Scalability of TTA

In this section, we describe the computational scaling properties and the accuracy of TTA in solving the smooth and non-smooth IRBC model. First, we increase the dimensionality of the model while keeping ranks and bases of the TTA fixed, and analyze the resulting computation times and errors. In a second step, we vary the ranks and bases in the non-smooth model while keeping the dimensionality fixed to assess their impact on accuracy.

Algorithm 2 Solving the IRBC Model using TTA

Require: Initial sample of states $\mathcal{S} = \{x_1, \ldots, x_N\} \subset X$, initial guess for next period's expectations terms $\eta^+ = (\eta_1^+, \ldots, \eta_M^+)$ in TTA format, error thresholds ε_{TI} and ε_{Sim}, sequence of shocks $\{z_t\}_{t=1}^T$, $\{\tilde{a}_t\}_{t=1}^T$, initial simulation state x_1.

Simulation & EGM

while $\xi_{\text{Sim}} > \varepsilon_{\text{Sim}}$ **do** ▷ Simulation Loop

  while $\xi_{\text{TI}} > \varepsilon_{\text{TI}}$ **do** ▷ EGM Loop

    for $k = 1, \ldots, N$ **do**

      At $x_k \in \mathcal{S}$, solve equations (16) and (17) for $\lambda(\tilde{x}_k), \tilde{k}_1, \ldots, \tilde{k}_M$ using EGM.

      Fix the choice $(k_1'(\tilde{x}_k), \ldots, k_M'(\tilde{x}_k)) = (k_1, \ldots, k_M) \subset x_k$.

      Given η^+, compute expectations over η^+ quasi analytically using (20).

      Obtain the endogenous state $\tilde{x}_k = (\tilde{k}, \tilde{a}_k, z_k)$ by rearranging equation (16).

      Compute policies $\eta(\tilde{x}_k) = (\eta_1(\tilde{x}_k), \ldots, \eta_M(\tilde{x}_k))$ via (19).

    end for

    Update $\eta_j := \mathcal{T}\left(m, r, \{x_k, \eta_j(x_k)\}_{k=1}^N\right)$, for $j = 1, \ldots, M$ using Algorithm 1.

    Set $\xi_{\text{TI}} = ||\eta - \eta^+||$ and update $\eta^+ := \eta$.

  end while

  Set $\tilde{\mathcal{S}} = \{\tilde{x}_1, \ldots, \tilde{x}_N\} := \mathcal{S}$.

  for $t = 1, \ldots, T$ **do** ▷ Update Simulation Path

    Evaluate η at x_t, infer k', λ using equations (19, 17), set $x_{t+1} := (k', \tilde{a}_{t+1}, z)$.

  end for

  Compute $\xi_{\text{Sim}} = ||\tilde{\mathcal{S}} - \mathcal{S}|| = \frac{1}{N} \sum_{i=1}^N |\tilde{x}_i - x_i|$.

end while

Draw new shocks $\{z_t\}_{t=1}^T$ and $\{\tilde{a}_t\}_{t=1}^T$.

for $t = 1, \ldots, T$ **do** ▷ Compute Euler Errors

  Evaluate η at x_t, and infer k', λ using equations (19, 17). Set $x_{t+1} := (k', \tilde{a}_{t+1}, z)$.

  Compute Euler Errors using (22) and (23).

end for

Scaling Exercise. In our first exercise, we scale the number of countries $M = 1, \ldots, 25$ in the (non-)smooth IRBC model, which implies dimensions $d = 2M + 1$, ranging from 11 to 51. We fix bases and ranks, $m = 3$ and $r = 3$, in the smooth model, for all dimensions; in the non-smooth specification we choose $m = 5$ and $r = 3$. The sample size $N = |\mathcal{S}|$ is chosen such that the ratio of sample points to tensor train parameters remains at 50, resulting in a sample size of $50mr^2d$. Figure 4 reports the normalized computation time — defined as the computation time of each model instance divided by that of the smooth five country model — and the (unit-free) error measure in percent for both models.The results show that computation time increases modestly and clearly sub-exponentially in the dimensionality of the problem.[14] The normalized scaling in Figure 4 clearly outperforms the adaptive sparse grids used by Brumm and Scheidegger (2017). The normalized computation time going from $d = 11$ to $d = 51$ increases modestly by a factor of 100. Computing the non-smooth model is not fundamentally more difficult than the smooth model, when accounting for the richer basis used to solve the non-smooth model. The average Euler error remains well below 0.001% (0.01%) in the smooth (non-smooth) model. In each case, the 99th-percentile error is approximately one order of magnitude higher than the average. In the non-smooth model the irreversibility constraint actually matters — the unconditional probability that a countries constraint is binding lies between 2 and 3 percent. The average error in the smooth model is roughly two orders of magnitude better than the simulation error reported in the scaling exercise of Brumm and Scheidegger (2017) in Figure 8, despite being orders of magnitude faster to compute.

The Role of Ranks and Bases. In the second exercise, we fix the dimension at $d = 7$ and study how approximation quality in the non-smooth model depends on the TTA rank and basis order. We increase the basis order m and rank r jointly while keeping the sample-to-parameter ratio fixed at 50. Moving from $m = 3$, $r = 3$ to $m = 5$, $r = 5$ lowers the average error from -4.85 to -5.02 and improves the 99th-percentile error from -4.19 to -4.30. Further refining ranks and basis yields diminishing returns: at $m = 9$, $r = 9$ the average error reaches -5.09 and the 99th-percentile error -4.37. Overall, the relatively small incremental gains at higher m and r suggest that even low ranks and low-order bases already capture the key features of the solution in this setting.

[14] Scaling is, however, more than linear. This is due to several contributing factors. First, to maintain a constant sample-to-parameter ratio, the sample size grows linearly in d. Second, the cost of evaluating the TTA at a given point in the state space grows linearly in d, at a rate of $\mathcal{O}(mr^2d)$. Together, this implies that the computational cost of a time iteration step (assuming a fixed number of Newton iterations) grows quadratically in d. Third, for each additional country, an additional policy function has to be approximated, while the approximation itself becomes more complicated due to the additional dimension and the larger sample size.

4 TTA for Heterogeneous Agents in Continuous Time

This section develops a tensor-train approach for solving heterogeneous agent models in continuous time. Section 4.1 presents the model and states the master equation characterizing equilibrium. Section 4.2 introduces an a posteriori model-reduction strategy for generating a discrete representation of the wealth distribution. Section 4.3 leverages that representation and the tensor train approach presented in Section 2 to build an algorithm for solving the master equation. Section 4.4 provides remaining algorithmic choices independent of the TTA approach. Section 4.5 defines an appropriate error metric and evaluates the method's performance, thereby demonstrating that the non-linearities induced by the stochastic wealth tax require a multi-dimensional representation of the wealth distribution to achieve accurate solutions.

4.1 Krusell-Smith Model with Stochastic Wealth Taxation

Households. The economy is populated by a continuum of infinitely lived heterogeneous households differing in idiosyncratic labor productivity $\varepsilon_t \in \varepsilon_1, \varepsilon_2$, discount rate $\rho_t \in \rho_1, \rho_2$ and endogenous asset holdings a_t. The idiosyncratic productivity state switches from j to the other state $\jmath$ with Poisson rate $\lambda_j > 0$, the idiosyncratic discounting state switches from i to $\imath$ with Poisson rate ω_i. Each household solves

$$\mathbb{E}_0 \int_0^\infty e^{-\rho_t t}\, u(c_t)\, dt, \quad \text{with} \quad u(c_t) = \frac{c_t^{1-\gamma} - 1}{1-\gamma}, \tag{24}$$

subject to

$$\dot{a}_t = w_t \varepsilon_t + r_t a_t - c_t, \quad a_t \geq \underline{a}, \tag{25}$$

with a borrowing limit $\underline{a} \geq 0$.

Production. A representative firm operates a Cobb-Douglas technology with aggregate capital K_t and inelastically supplied labor $N_t \equiv 1$, so that output, the wage, and the interest rate are given by:

$$Y_t = \exp(z_t) K_t^\alpha, \quad w_t = (1-\alpha)\exp(z_t) K_t^\alpha, \quad r_t = \alpha \exp(z_t) K_t^{\alpha-1} - \delta.$$

The (log) total factor productivity z_t follows an Ornstein-Uhlenbeck process in levels

$$dz_t = \kappa(\bar{z} - z_t)dt + \sigma_z dB_t, \tag{26}$$

with mean reversion $\kappa > 0$, long-run mean $\bar{z}$, and diffusion σ_z. Aggregate capital equals the first moment of the cross-sectional wealth distribution,

$$K_t = \int_{\underline{a}}^{\infty} a \sum_{i=1}^{2} \sum_{j=1}^{2} g_{ij}(a,t) da, \tag{27}$$

where $g_{ij}(a,t)$ denotes the density of agent with assets a, productivity type j, and discounting type i.

Wealth Tax. With Poisson intensity θ, a stochastic wealth-tax is levied. At such an event, the government collects a fraction $\tau \in [0,1)$ of each agent's assets and distributes the proceeds equally across agents, yielding post-tax asset holdings $\tilde{a}$ from pre-tax asset holdings a satisfying:

$$\tilde{a} = (1-\tau)a + \tau K_t. \tag{28}$$

HJB Equation. The value function for type ij, $V^{ij}(a,t)$, satisfies the HJB equation:

$$\begin{aligned}\rho_i V^{ij}(a,t) = \max_c \Big\{ & u(c) + \partial_a V^{ij} \left[w_t \varepsilon_j + r_t a - c \right] + \lambda_j \left[V^{ij} - V^{ij} \right] \\ & + \omega_i \left[V^{\iota j} - V^{ij} \right] + \partial_t \mathbb{E}_t \{ V^{ij} \} \Big\}\end{aligned}$$

The first-order condition implies

$$c_{ij}(a,t) = u'^{-1}\left(\partial_a V^{ij}(a,t) \right), \quad s_{ij}(a,t) = w_t \varepsilon_j + r_t a - c_{ij}(a,t), \tag{29}$$

where $s_{ij}(a,t)$ denotes the optimal savings function (drift of assets) for type ij.

Kolmogorov Forward Equation. Given optimal savings $s_{ij}(a,t)$, the law of motion for the density function of type ij is given by the Kolmogorov forward equation (KFE):

$$\begin{aligned}\partial_t g_{ij}(a,t) = & -\partial_a [s_{ij}(a,t) g_{ij}(a,t)] - \lambda_j g_{ij}(a,t) + \lambda_j g_{i\jmath}(a,t) - \omega_i g_{ij}(a,t) + \omega_\iota g_{\iota j}(a,t) \\ & + \mathbb{1}_\theta \left[\frac{1}{1-\tau} g_{ij}\left(\frac{a - \tau K_t}{1-\tau}, t \right) - g_{ij}(a,t) \right],\end{aligned}$$

i.e., transport by savings, switching between idiosyncratic states, and a jump (push-forward) operator for the tax (where $\mathbb{1}_\theta$ indicates a tax event).[15] The borrowing con-

[15] The jump term consists of an outflow $-g_j(a,t)$, and an inflow from the pre-image of the post-tax mapping. The additional term $\frac{1}{1-\tau}$ comes from a mass conservation consideration.

straint induces a reflecting boundary at $a = \underline{a}$ (zero outward flux). For compactness, we write $\partial_t g = (\mathcal{A}^* g)$.

Distributional Approximation. To make the infinite-dimensional state tractable, we summarize the wealth distribution by M moments $\Gamma_t = (m_t^1, \ldots, m_t^M)$ with

$$m_t^k = \int_0^\infty \sum_{i=1}^{2} \sum_{j=1}^{2} f_{ij}^k(a) g_{ij}(a,t) da \qquad k = 1, \ldots, M, \tag{30}$$

for a chosen set of functions f_{ij}^k. For example, taking $f_{ij}^1(a) = a$, for $i = 1, 2$ and $j = 1, 2$ yields the capital stock.

Recursive Equilibrium. Turning from sequential to recursive characterization of the model, policy and value functions now depend on the distribution over endogenous assets and exogenous (productivity and discounting) types — or a finite dimensional representation of it, as just introduced. More precisely, a recursive competitive equilibrium consists of functions $\{V^{ij}, g^{ij}, c^{ij}, s^{ij}\}(a, z, \Gamma)$ for $i = 1, 2$, $j = 1, 2$ and aggregates $\{r, w, Y, K, C\}(z, \Gamma)$ such that (i) households optimize given rational expectations, (ii) factor prices are competitive, (iii) aggregates are consistent with distributions, and (iv) the asset distribution evolves according to the Kolmogorov forward equation given policies.

Master Equation. The recursive-equilibrium value function satisfies the following master equation, which can be derived by combining the HJB and Kolmogorov forward equation:

$$\begin{aligned}
\rho_i V^{ij}(a, z, \Gamma) = \max_c \Big\{ & u(c) + \partial_a V^{ij} \left[w(z, \Gamma) \varepsilon_j + r(z, \Gamma) a - c \right] \Big\} \\
& + \lambda_j \big(V^{ij} - V^{ij} \big) + \omega_i \left(V^{\imath j} - V^{ij} \right) \\
& + \kappa(\bar{z} - z)\, \partial_z V^{ij} + \tfrac{1}{2} \sigma_z^2\, \partial_{zz} V^{ij} \\
& + \mu_\Gamma(\Gamma) \cdot \nabla_\Gamma V^{ij} + \theta \big(V^{ij}(\tilde{a}, z, \tilde{\Gamma}) - V^{ij}(a, z, \Gamma) \big),
\end{aligned}$$

where $\mu_\Gamma(\Gamma)$ is the drift of the chosen moment vector induced by the KFE and $\tilde{\Gamma}$ denotes the post-tax moments implied by $\tilde{a}$. At the borrowing limit $a = \underline{a}$, the drift of assets must be non-negative implying the following boundary condition

$$\partial_a V^{ij}(\underline{a}, \cdot) \geq u' \left(w(z, \Gamma) \varepsilon_j + r(z, \Gamma) \underline{a} \right). \tag{31}$$

Table 2: Calibration of the Continuous-Time Heterogeneous-Agent Economy.

Parameter	Symbol	Value
Discount rate	$[\rho_1, \rho_2]$	$[0.0, 0.09]$
Risk aversion	γ	2.0
Capital share	α	0.36
Depreciation	δ	0.02
Borrowing limit	$\underline{a}$	0.0
Labor productivity	$[\underline{\varepsilon}, \bar{\varepsilon}]$	[0.15, 1.096]
Productivity switching rates	$[\lambda_1, \lambda_2]$	[0.4, 0.045]
Share of high discounting households	d	0.5
Discounting redraw	q	0.05
Discounting switching rates	$[\omega_1, \omega_2]$	$[(1-d)q, dq]$
TFP long-run mean	$\bar{z}$	0.0
TFP diffusion (OU)	σ_z	0.007
OU mean reversion	κ	0.05

Calibration. We follow Krusell and Smith (1998) for the idiosyncratic labor-productivity process, aggregate production technology, and risk aversion. In addition, we introduce two discount-factor types, which increases mass at the borrowing constraint and in the upper tail of the wealth distribution. The discount factors ρ_1 and ρ_2 are set to the values in Auclert et al. (2025). We assume on average half of households has patience ρ_1 or ρ_2, respectively, and with a probability of 5% per quarter patience is redrawn. The continuous TFP process is calibrated to the empirical estimates in Christensen et al. (2024). Parameter values are reported in Table 2.

4.2 A Posteriori Model Reduction for Distributional Dynamics

We now propose a novel, yet straightforward, simulation-based method to identify moments for a low-dimensional representation of the wealth distribution.

Dynamic Mode Decomposition. Let $g(t) \in \mathbb{R}^N$ be a finite-dimensional representation of the wealth distribution (e.g., histogram bins). To compute informative moments of the distribution, we assume its evolution is approximately linear, with forward operator A and an error term $\varepsilon(t)$

$$\dot{g}(t) \;=\; A\, g(t) + \varepsilon(t), \qquad A \in \mathbb{R}^{N\times N},\ \varepsilon(t) \in \mathbb{R}^N.$$

We observe $g(t)$ at equally spaced times $t_1, \ldots, t_T$ with step Δt, and collect snapshot matrices[16]

$$X_- \;=\; \big[g(t_1), \ldots, g(t_{T-1})\big] \in \mathbb{R}^{N\times(T-1)}, \qquad X_+ \;=\; \big[g(t_2), \ldots, g(t_T)\big] \in \mathbb{R}^{N\times(T-1)}.$$

Dynamic Mode Decomposition (DMD) seeks the best linear propagator A_d (in least squares) such that $X_+ \approx A_d\, X_-$. When N is large, we compute a rank-k truncated singular value decomposition (SVD) of X_-,

$$X_- \;\approx\; U\,\Sigma\,V^\top, \qquad U \in \mathbb{R}^{N\times k},\ \Sigma \in \mathbb{R}^{k\times k},\ V \in \mathbb{R}^{(T-1)\times k},$$

where $\Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_k)$ collects the k largest singular values. The columns of U span the dominant directions of variation in the data, and the rows of $V^\top$ capture their time paths. Projecting onto U yields compact "moment" coordinates

$$m(t) \;=\; U^\top g(t) \in \mathbb{R}^k, \qquad m(t_{i+1}) \;\approx\; \tilde{A}\, m(t_i),$$

with reduced dynamics estimated by

$$\tilde{A} \;=\; U^\top X_+\, V\,\Sigma^{-1} \;\in\; \mathbb{R}^{k\times k}.$$

These moments $m(t)$ are the variables we carry into the global solution algorithm to represent the distribution parsimoniously.[17] As a heuristic for choosing how many moments to retain, we use the energy share of each singular value,

$$E_i = \frac{\sigma_i^2}{\sum_j \sigma_j^2}, \tag{32}$$

which quantifies the fraction of variance explained by the i-th singular value.

Connection to Koopman Theory. When the distribution follows nonlinear dynamics $g(t+\Delta t) = f\big(g(t)\big)$, the Koopman operator $\mathcal{K}$ advances observables ψ linearly:

$$\frac{d}{dt}\psi\big(g(t)\big) \;=\; \mathcal{K}_c\,\psi\big(g(t)\big)$$

[16] We obtain $g(t)$ by simulating an approximate model solution (e.g., through histogram evolution).

[17] We make two practical adjustments to the otherwise canonical model reduction approach. First, since the distributional dynamics also depend on productivity z_t, we also include it into the forecasting rule. Second, due to its importance for factor prices, we use capital as the first moment. To ensure that higher order moments are orthogonal to capital, the matrix X_- is projected onto the subspace orthogonal to the capital weight vector before performing the SVD.

Thus, even if f is nonlinear in state space, evolution is linear in the (typically infinite-dimensional) space of observables. DMD can be viewed as a finite-dimensional approximation to $\mathcal{K}$ restricted to linear observables $\psi(g) = g$. Extended DMD (EDMD) enlarges the observable set to a dictionary

$$\psi(g) \;=\; \big[\phi_1(g), \ldots, \phi_M(g)\big]^\top$$

(e.g., polynomials or radial basis functions) and fits a linear operator $\widehat{\mathcal{K}}$ from

$$\Psi_- = \big[\psi(g(t_1)), \ldots, \psi(g(t_{T-1}))\big], \qquad \Psi_+ = \big[\psi(g(t_2)), \ldots, \psi(g(t_T))\big]$$

via the relation $\Psi_+ \approx \widehat{\mathcal{K}}\,\Psi_-$. With a sufficiently rich dictionary, EDMD captures nonlinear distribution dynamics through a linear evolution in feature space, while still delivering a low-dimensional set of learned moments for the global solution.

4.3 Solving the Master Equation with Tensor Trains

We solve the master equation using value function iteration (VFI). In each VFI step, we recover optimal consumption c^* from the first-order condition, taking the current candidate value function V as given. Conditional on c^*, we update V by approximately solving the master equation with a spectral least squares method.[18] The solver operates directly on tensor-train (TT) coefficients, which allows the high-dimensional update to be carried out efficiently (see Section 2).

A generic master-equation problem has two components: (i) an interior equation that must hold on the state space, and (ii) boundary conditions. We compute an approximate solution by casting the problem as constrained least squares, in the spirit of Section 2. Specifically, we (a) minimize the squared residual of the interior equation evaluated at a set of interior points, and (b) enforce the boundary conditions at a separate set of boundary points.

The boundary conditions typically take the form of inequality constraints. For computational convenience, we convert each inequality constraint into an equality constraint by introducing a nonnegative slack variable at each boundary point. We then solve the resulting equality-constrained problem using an augmented Lagrangian approach, which incorporates the constraints into the objective via multipliers and quadratic penalties. Given this structure, the augmented-Lagrangian subproblems admit closed-form updates for both the value-function coefficients and the slack variables. We obtain a solution by iterating over updates of (i) the value-function coefficients, (ii) the slack variables, and (iii) the Lagrange multipliers — that is, by applying

[18] By "spectral least squares" we mean a global (typically polynomial) basis representation, with the differential equations enforced in a least-squares sense.

the alternating direction method of multipliers (ADMM). One run of ADMM corresponds to a single VFI update.

Updating the value-function coefficients amounts to a quadratic minimization problem. In TT form, this step can be solved efficiently using alternating least squares (ALS), which preserves the low-rank structure throughout.

This section proceeds in four steps. First, we introduce a generic master-equation problem and its constrained least-squares formulation. Second, we derive the corresponding augmented-Lagrangian representation. Third, we map our application from the previous section into this general framework. For clarity, we present the full-tensor formulation in the main text; the local subproblems underlying the TT implementation (TTA) are deferred to the appendix. Finally, we describe the ADMM iterations that implement one VFI update.

Generic Master Equation. Let $\mathcal{D}$ be the state domain with interior $\mathrm{int}(\mathcal{D})$ and boundary $\partial\mathcal{D}$. Consider master equations that act linearly on V in the interior and on the boundary. Write $\mathcal{M}$ for the interior operator and $\mathcal{C}$ for the boundary operator. We seek a TT approximation $V(\cdot, \mathcal{T})$ with cores $\mathcal{T} = \{W^1, \ldots, W^d\}$ satisfying

$$\mathcal{M}[V(\cdot;\mathcal{T})](x) = u(x), \quad \forall x \in \mathrm{int}(\mathcal{D}), \quad \mathcal{C}[V(\cdot;\mathcal{T})](x) \geq h(x), \quad \forall x \in \partial\mathcal{D}.$$

Global Minimization Problem. Given interior samples $\{x_i\}_{i=1}^N$ and boundary samples $\{\bar{x}_\ell\}_{\ell=1}^M$, we minimize interior residuals with regularization for numerical stability and impose boundary inequalities

$$\begin{aligned} \min_{W_1,\ldots,W_d} \quad & \sum_{i=1}^{N} ||\mathcal{M}[V(\cdot;\mathcal{T})](x_i) - u(x_i)||_2^2 + \eta \sum_{j=1}^{d} ||W_j||_F^2 \\ \text{s.t.} \quad & \mathcal{C}[V(\cdot;\mathcal{T})](\bar{x}_\ell) \geq h(\bar{x}_\ell), \qquad \ell = 1, \ldots, M. \end{aligned}$$

Augmented Lagrangian Problem. To cast the constraint minimization above into, an unconstrained convex problem we proceed in two steps: First, we introduce slack variables $t_\ell \geq 0$ at boundary samples $\{\bar{x}_\ell\}_{\ell=1}^M$. Second, we incorporate the constraints

into the objective via multipliers and quadratic penalties — an approach known as augmented Lagrangian:

$$\begin{aligned}\mathcal{L}(\mathcal{T}, t, \mu) = &\sum_{i=1}^{N} ||\mathcal{M}[V(\cdot;\mathcal{T})](x_i) - u(x_i)||_2^2 + \eta \sum_{j=1}^{d} ||W_j||_F^2 \\ &+ \frac{\gamma}{2} \sum_{\ell=1}^{M} \left(\mathcal{C}[V(\cdot;\mathcal{T})](\bar{x}_\ell) - h(\bar{x}_\ell) - t_\ell\right)^2 \\ &+ \sum_{\ell=1}^{M} \mu_\ell \left(\mathcal{C}[V(\cdot;\mathcal{T})](\bar{x}_\ell) - h(\bar{x}_\ell) - t_\ell\right), \quad t_\ell \geq 0.\end{aligned}$$

Full Tensor Problem. Given multipliers μ and slack variables t write the minimization problem over $\mathcal{T}$, as problem over the full tensor of value function coefficients $v = \text{vec}(A) \in \mathbb{R}^K$ with $K = \prod_{j=1}^{d} m_j$ as[19]

$$\min_v \frac{1}{2}||Mv - u||_2^2 + \frac{\gamma}{2}||Cv - h - t||_2^2 + \mu^\top (Cv - h - t). \tag{33}$$

This problem falls into the class of quadratic minimization problems that can be solved efficiently using ALS. In Appendix C we show how to solve the minimization problem in TTA form, for ease of exposition we stick with the full tensor problem here. The component matrices M, C and the vectors u, h take the following form in the application outlined above:

$$\begin{aligned}M_{ij} = &\left(\rho_i + \frac{1}{\Delta} + \lambda_j + \omega_i + \theta\right) X^{V,ij} - \left(w(z,\Gamma)\varepsilon_j + r(z,\Gamma)a - c_{ij}\right) X^{\partial_a,ij} \\ &- \kappa(\bar{z} - z) X^{\partial_z,ij} - \frac{1}{2}\sigma_z^2 X^{\partial_z^2,ij} - \mu(\Gamma) X^{\partial_\Gamma,ij} - \lambda_j X^{V,ij} - \omega_i X^{V,ij} - \theta X^{\text{tax},ij},\end{aligned}$$

$$u_{ij} = u(c^{ij}) + \frac{1}{\Delta} V^{ij,\text{old}}, \quad C_{ij} = X^{\partial_a,ij}, \quad h_{ij} = u'(w(z,\Gamma)\varepsilon_j + r(z,\Gamma)\underline{a}).$$

where $X^{\square,ij}$ is the full matrix of basis entries of the value function, or its derivatives.[20]

ADMM. We obtain a solution of the minimization problem by iterating over updates of the value-function coefficients, the slack variables and the multipliers, of the augmented Lagrangian — this approach is known as alternating directions method of

[19] With $M \in \mathbb{R}^{N\times K}$, $u \in \mathbb{R}^N$, $C \in \mathbb{R}^{M\times K}$, $h \in \mathbb{R}^M$, $\mu \in \mathbb{R}^M$, $t \in \mathbb{R}^M$. In practice we also add a small regularization term to each core W^k for numerical stability.

[20] For example $X^{V,ij} = \begin{bmatrix} \{\phi_{i_1}(x_1^1) \cdot \ldots \cdot \phi_{i_d}(x_d^1)\}_{i_1=1,\ldots,i_d=1}^{m_1,\ldots,m_d} \\ \vdots \\ \{\phi_{i_1}(x_1^N) \cdot \ldots \cdot \phi_{i_d}(x_d^N)\}_{i_1=1,\ldots,i_d=1}^{m_1,\ldots,m_d} \end{bmatrix} \in \mathbb{R}^{N\times K}$.

multipliers (ADMM). One iteration consists of minimizing the augmented Lagrangian with respect to v (in TT format)

$$\mathcal{T}^{n+1} := \arg\min_{v} \frac{1}{2}||Mv - u||_2^2 + \frac{\gamma}{2}||Cv - h - t^n||_2^2 + \mu^{n\top}(Cv - h - t^n),$$

which admits a closed form least squares solution. Followed by updating the slack variables, by minimizing the augmented Lagrangian with respect to t,

$$t^{n+1} = \max\left\{\mathcal{C}[V(\cdot;\mathcal{T}^{n+1})](\bar{x}) - h(\bar{x}) + \frac{\mu^n}{\gamma}, 0\right\}, \tag{34}$$

also with a closed form solution. Lastly, in the dual ascent step we update the multiplier, as is standard in ADMM

$$\mu^{n+1} = \mu^n + \gamma\left(\mathcal{C}[V(\cdot;\mathcal{T}^{n+1})](\bar{x}) - h(\bar{x}) - t^{n+1}\right). \tag{35}$$

Once the ADMM algorithm converged, the value function iteration step is complete. We adapt the penalty γ to balance primal and dual progress.[21] After completing the VFI step, previous slacks, multipliers and penalties can be used to warm start subsequent iterations.

4.4 Solution Algorithm

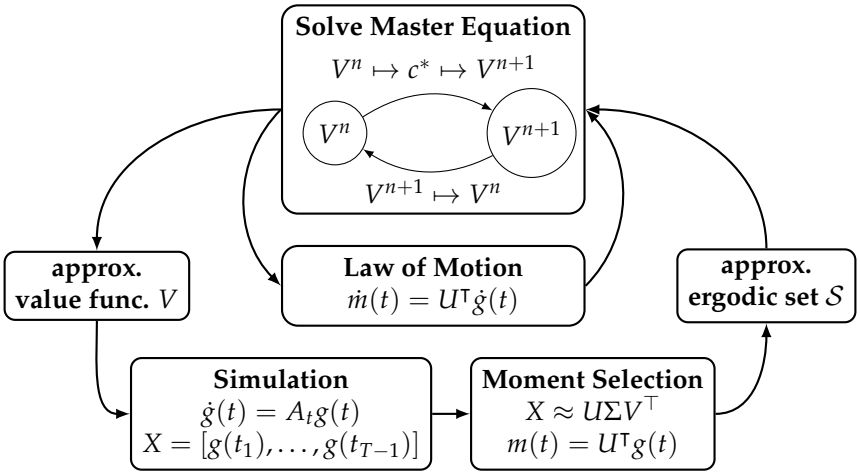
The solution algorithm consists of three nested routines. At the outermost level, we construct a training set: a collection of histogram snapshots of the cross-sectional distribution paired with realizations of the exogenous shocks. At the same level, we define a set of moments, represented as weight vectors that map a histogram snapshot into a scalar statistic. Given a training set and a choice of moments, solving the household problem requires a forecasting rule for the evolution of the aggregate state (the moments). These forecasting rules are determined at the second level. At the innermost level, given moments and forecasting rules, we solve the household problem by value function iteration (VFI). This step is taken as given since it was described in the previous section. A graphical overview of these three nested procedures is provided in Figure 5.

Training Set and Moments. We construct the training set from simulated data obtained by simulating forward the distribution under the exogenous shocks and a candidate solution for the household value function.[22] From the simulated sample we

[21] We choose γ such that $||r_{\text{pri}}||_2$ and $||r_{\text{dual}}||_2$ are of comparable magnitude.

[22] To obtain an initial simulation set, we first solve the household problem under an ad hoc forecasting rule and simulate the model under this rough approximation.

Figure 5: Solution Algorithm for Heterogeneous Agent Model



The algorithm has three nested loops. The outer loop builds a training set of histogram snapshots of the cross-sectional distribution paired with exogenous shock realizations, and specifies a set of moments as weight vectors that map each histogram to scalar statistics. Given a training set and chosen moments, the middle loop estimates forecasting rules for the evolution of the aggregate state (the moments). In the inner loop, taking these moments and forecasting rules as given, the household problem is solved via value function iteration with TTA.

select moments using the a posteriori model reduction procedure described above. We then draw training points by randomly sampling tuples consisting of (i) the exogenous shocks, (ii) histogram snapshots, and (iii) time derivatives of histogram snapshots. Given the moment definitions, we compute the implied moments at each training point.

Forecasting Rules. To solve the household problem, agents require beliefs about the evolution of the distribution. We represent these beliefs in terms of the implied evolution of the moments. Since the training data include time derivatives of the histogram snapshots, we can compute time derivatives of moments directly. We estimate an approximate law of motion by regressing the time derivatives of moments on current moments and exogenous states. In addition, households must forecast how an arrival of the wealth-tax shock reshapes the distribution in moment space. We therefore estimate a separate mapping from current moments (and exogenous states) into post-shock moments.

Overall Algorithm. Given these components, the algorithm proceeds as follows. On the innermost level, we solve the household problem using VFI and ADMM. After the value function has converged, we update the forecasting rules implied by the new household solution. Once the value function and forecasting rules are jointly consistent, we simulate the model forward. Based on the resulting simulated sample, we (re-)select moments via a posteriori model reduction and update the training set. We

iterate until the training set, the selected moments, the forecasting rules, and the household value function are mutually consistent on the implied ergodic set. Algorithm 4 in Appendix C summarizes the full procedure.

4.5 Measuring Accuracy when Aggregate Risk Matters

In this section, we show that the Krusell–Smith model with large wealth-tax shocks requires a genuinely multidimensional representation of the wealth distribution, and that the tensor-train approximation (TTA) enables us to solve the resulting high-dimensional PDE accurately. We begin by introducing our error metric, and then demonstrate that this metric improves markedly as we enrich the state representation with additional moments.

Error Measure. We evaluate accuracy via a relative HJB error in consumption units. Given a candidate value function $V^{ij}(\cdot;\mathcal{T})$, obtain the optimal consumption from the FOC,

$$c^*_{ij}(a,z,\Gamma) = (u')^{-1}\left(\partial_a V^{ij}(a,z,\Gamma;\mathcal{T})\right).$$

Compute the implied consumption $\tilde{c}_{ij}(a,z,\Gamma)$ from the master equation

$$\begin{aligned}\tilde{c}^*_{ij}(a,z,\Gamma) = u^{-1}\Big(&\rho_i V^{ij}(a,z,\Gamma;\mathcal{T}) - \partial_a V^{ij}(a,z,\Gamma;\mathcal{T})\left[w(z,\Gamma)\varepsilon_j + r(z,\Gamma)a - c^*_{ij}\right]\\ &- \lambda_j\big(V^{ij}(a,z,\Gamma;\mathcal{T}) - V^{ij}(a,z,\Gamma;\mathcal{T})\big)\\ &- \omega_i\left(V^{ij}(a,z,\Gamma;\mathcal{T}) - V^{ij}(a,z,\Gamma;\mathcal{T})\right)\\ &- \kappa(\bar{z}-z)\,\partial_z V^{ij}(a,z,\Gamma;\mathcal{T}) - \tfrac{1}{2}\sigma_z^2\,\partial_{zz}V^{ij}(a,z,\Gamma;\mathcal{T})\\ &- \mu_\Gamma(\Gamma)\cdot\nabla_\Gamma V^{ij}(a,z,\Gamma;\mathcal{T}) - \theta\big(V^{ij}(\tilde{a},z,\tilde{\Gamma};\mathcal{T}) - V^{ij}(a,z,\Gamma;\mathcal{T})\big)\Big)\end{aligned}$$

At the borrowing constraint, we also record the boundary violation in relative consumption units,

$$\mathcal{E}_B(a,z,\Gamma) = \begin{cases} 0, & a > \underline{a}\\ \frac{\max\left\{c_{\min}(z,\Gamma) - (u')^{-1}\left(\partial_a V^{ij}(\underline{a},z,\Gamma;\mathcal{T}),0\right)\right\}}{c_{\min}(z,\Gamma)}, & a = \underline{a}\end{cases}$$

Our pointwise error metric is then

$$\mathcal{E}_{ij}(a,z,\Gamma) = \max\left\{\left|\frac{\tilde{c}(a,z,\Gamma)}{c^*(a,z,\Gamma)} - 1\right|, \mathcal{E}_B(a,z,\Gamma)\right\}.$$

We simulate a path of aggregate distributions $\{g(t)\}_{t=1}^T$ by drawing shocks and evolving the Kolmogorov forward equation on a fine wealth grid using the policies implied

by $V^{ij}(\cdot;\mathcal{T})$. At each t, we draw 100 individuals from the joint distribution over (i,j) and assets $a \sim g_{ij}(t)$, evaluate $\mathcal{E}_{ij}(a,z,\Gamma_t)$ at those draws, and summarize the resulting error distribution over time.

Accuracy Results. To demonstrate that TTA can accurately solve multidimensional heterogeneous agent models, we solve the economy under three tax regimes: 0% (no tax), 10%, and 20% wealth tax, each arriving with quarterly probability of 2.5%. For each regime we vary the number of aggregate moments, M, used to summarize the cross-section. We approximate the value function (for each discrete idiosyncratic state) with TTA over the continuous state vector $(a, z, k, m_2, \ldots, m_M)$.[23] For assets, a, we use a basis of order 30, for each additional dimension we use a basis of order 3, while setting all TT ranks equal to 3.

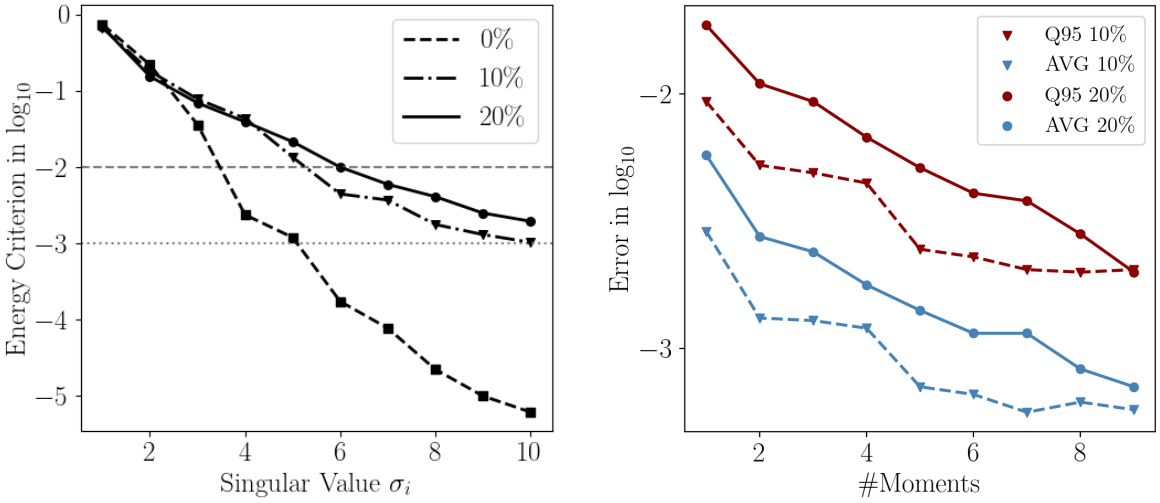
Figure 6 reports statistics from solving the model under different tax regimes and with a varying number of aggregate moments. On the left-hand side of that figure, we report the energy shares E_i of the singular values from the model-reduction step. Without a wealth tax, E_i declines steeply, indicating that higher moments add little to the aggregate dynamics. With a 10% or 20% wealth tax, the decay is notably flatter: higher moments contribute meaningfully and are therefore required to capture the dynamics. The drop from 0% to 10% is much larger than from 10% to 20% — even a 10% tax, occurring on average every ten years, already reshapes the wealth distribution and raises the model's effective dimensionality, while the incremental effect of a larger tax is comparatively smaller.

On the right-hand side of Figure 6 we plot the mean error and the 95th-percentile error (both on log-10 scale) against the number of moments used to summarize the distribution. For both positive tax levels and both error metrics, accuracy improves roughly by an order of magnitude as additional moments are included. In the 10% case, gains taper after the fifth moment, whereas the 20% case continues to benefit from additional moments — consistent with the flatter energy profile under the higher tax. At a low number of moments, the 10%-tax economy is more accurate than the 20%-tax economy, but the gap narrows as the number of moments grows.[24]

[23] Capital is fixed as the first moment because it directly pins down factor prices; the remaining moments are selected by a posteriori model reduction. To avoid redundancy once capital is fixed as the first moment, we orthogonalize the remaining moment time series with respect to the first moment (i.e., remove the first-moment component). All results use 10,000 sample points and the linear DMD specification described in the previous section.

[24] In the tax-free economy errors are substantially lower (-3.9 for the mean error and -3.5 at the 95th percentile), as the tax introduces discrete jumps and more distributional movement, while the TFP process alone generates relatively little variation.

Figure 6: Heterogeneous Agent Model — Energy Shares and Errors.



The left panel shows the energy shares E_i of the i-th singular value (index reported on the horizontal axis) from the model-reduction step under wealth-tax rates of 0%, 10%, and 20%. Taxation flattens the singular-value decay, indicating a higher effective dimensionality than in the no-tax economy. The right panel plots the mean and 95th-percentile master equation errors against the number of aggregate moments employed in the solution — adding moments yields substantial accuracy gains.

5 Conclusion

We develop a novel method for computing equilibria in dynamic economic models using the tensor train decomposition. By exploiting latent low-rank structures in policy or value functions, the approach delivers accurate approximations while remaining computationally tractable in high-dimensional state spaces.

We first validate performance in a smooth and non-smooth international real business cycle model, assessing approximation quality and computational scaling across dimensions. The results show that accuracy can be preserved even as the number of state variables increases, with runtimes rising only moderately. Beyond scalability, the TT representation facilitates fast and reliable expectation computation, supports flexible training-sets targeted to the ergodic set, and makes it possible to implement the endogenous grid method in high dimensions.

We then extend the framework to heterogeneous-agent settings, solved in continuous time via a spectral least-squares tensor-train approach. In a Krusell–Smith economy with a stochastic wealth tax, we demonstrate that TTA can solve the high-dimensional master equation accurately and, crucially, that capturing multiple distributional moments is necessary to achieve accurate solutions.

Overall, tensor-based methods provide a scalable and reliable solution strategy for high-dimensional economic models. TTA offers a flexible toolkit for future applications in quantitative macroeconomics and beyond.

APPENDIX

A Generic ALS Template

The TTA algorithm introduced in Section 2 is applicable to a wide range of quadratic problems that can be minimized efficiently using ALS. Let $A \in \mathbb{R}^{m_1 \times \ldots \times m_d}$ be the full tensor $u = \mathrm{vec}(A) \in \mathbb{R}^M$ with $M = \prod_{k=1}^{d} m_k$. We consider functionals with constant global Hessian

$$\mathcal{J}(u) = \tfrac{1}{2}||Su - b||_2^2 + \langle Bu - c, e \rangle \,, \tag{36}$$

where $S \in \mathbb{R}^{N \times M}, b \in \mathbb{R}^N, B \in \mathbb{R}^{K \times M}, c \in \mathbb{R}^K$ and $e \in \mathbb{R}^K$. This class of problems covers the least-squares approximation problem (Algorithm 1), the augmented-Lagrangian objective in Section 4, and others; see Holtz et al. (2012). If a problem can be posed as (36) and all but one TT cores are frozen, each ALS update reduces to solving a small linear system, which makes this class particularly attractive for TTA.

Local Normal Equations. Let $W^k \in \mathbb{R}^{r_{k-1} \times m_k \times r_k}$ be the k-th core and $v = \mathrm{vec}(W^k) \in \mathbb{R}^{r_{k-1} m_k r_k}$. With all other cores frozen, the vectorized tensor can be written as

$$u = P_k v,$$

where $P_k \in \mathbb{R}^{M \times r_{k-1} m_k r_k}$ is the linear retraction operator that maps the local core into the global tensor. Then the local minimization $\min_v \mathcal{J}(P_k v)$ has the normal equations

$$\hat{S}_k^\top \hat{S}_k v = \hat{S}_k^\top b - \hat{B}_k^\top e, \quad \hat{S}_k := S P_k, \quad \hat{B}_k := B P_k \tag{37}$$

These are small linear systems of size $r_{k-1} m_k r_k$.

Efficient Implementation via Environments. Algorithm 1 introduced left / right stacks. They provide an efficient implementation of the reduced objects $\hat{S}_k$ and $\hat{B}_k$, so neither P_k nor S is ever formed explicitly.

Least-Squares Special Case. For the dataset $\{x^n, y^n\}_{n=1}^N$ with basis $\{\phi_{i_j}(\cdot)\}_{i_j=1}^{m_j}$ in each dimension j, the full-tensor least-squares problems corresponds to $B = 0, c = 0, e = 0$, and[25]

$$S = \begin{bmatrix} \left\{\prod_{j=1}^d \phi_{i_j}(x_j^1)\right\}_{i_1=1,\dots,i_d=1}^{m_1,\dots,m_d} \\ \vdots \\ \left\{\prod_{j=1}^d \phi_{i_j}(x_j^M)\right\}_{i_1=1,\dots,i_d=1}^{m_1,\dots,m_d} \end{bmatrix} \in \mathbb{R}^{N\times M}, \quad b = \begin{bmatrix} y^1 \\ \vdots \\ y^N \end{bmatrix} \in \mathbb{R}^N. \tag{38}$$

Let $\mathcal{L}^k \in \mathbb{R}^{N\times r_{k-1}}$ and $\mathcal{R}^k \in \mathbb{R}^{N\times r_k}$ denote the left/right stacks defined recursively from the cores and basis evaluations for all points in the dataset. Then $\hat{S}_k$ admits the compact factorization

$$\hat{S}_k = \mathcal{L}^k_{nj_{k-1}} \cdot \phi(x_k)_{ni_k} \cdot \mathcal{R}^k_{nj_k} \in \mathbb{R}^{N\times r_{k-1}m_k r_k}, \tag{39}$$

without ever building S or P_k. It is this structure that mitigates the curse of dimensionality in the ALS update steps. Algorithm 3 provides a generic formulation of ALS.

B Details Non-Smooth IRBC Model

This appendix details the non-smooth variant of the IRBC model introduced in Section 3. In contrast to the smooth benchmark, investment is irreversible:

$$k'_j - (1-\delta)k_j \geq 0, \qquad j = 1, \dots, M. \tag{40}$$

The aggregate resource constraint (17) is unchanged. The Euler condition for capital now carries the KKT multiplier μ_j associated with (40):

$$\lambda[1 + \phi g_j] - \mu_j = \beta \mathbb{E}_t \left\{ \lambda' \left[a'_j A \zeta (k'_j)^{\zeta-1} + 1 - \delta + \frac{\phi}{2} g'_j (g'_j + 2) \right] - (1-\delta)\mu'_j \right\}. \tag{41}$$

The multiplier satisfies the complementarity conditions

$$0 \leq u_j \perp (k'_j - k_j(1-\delta)) \geq 0. \tag{42}$$

For later shorthand, define the country-specific expectation kernel

$$\eta_j(x') = \lambda' \left[a'_j A \zeta (k'_j)^{\zeta-1} + 1 - \delta + \frac{\phi}{2} g'_j (g'_j + 2) \right] - (1-\delta)\mu'_j. \tag{43}$$

[25] We have excluded the regularization term, which is only for numerical stability at the local cores.

Algorithm 3 Generic ALS Algorithm

Require: Basis dimensions $m = (m_1, \ldots, m_d)$; ranks $r = (r_0, \ldots, r_d)$ with $r_0 = r_d = 1$; initial right-orthonormal cores $W^1, \ldots, W^d$ with $W^k \in \mathbb{R}^{r_{k-1} \times m_k \times r_k}$; operator $\mathcal{J}$ of the form (36) with $S \in \mathbb{R}^{N \times M}, b \in \mathbb{R}^N, B \in \mathbb{R}^{K \times M}$, and $c \in \mathbb{R}^K, e \in \mathbb{R}^K$.

Generic Alternating Least Squares

Set $\tilde{\xi} = \infty$ and $\epsilon = \infty$.

while $\epsilon > \varepsilon$ **do** ▷ Macro-Iterations (sweeps)

for $k = 1, \ldots, d-1$ **do** ▷ First Half-Sweep

Solve $v^k := \arg\min_v \mathcal{J}(P_k v)$ via (37) using $\hat{S}_k$ and $\hat{B}_k$.

Compute $[Q, R] = \mathrm{qrd}(\mathrm{unf}_L(\mathrm{res}(v^k)))$.

Set $W^k := \mathrm{res}(Q), W^{k+1} := \mathrm{res}\left(\mathrm{unf}_R\left(W^{k+1}\right) \cdot R^\top\right)$.

Update left stack $\mathcal{L}^{k+1}$ using equation (7).

end for

for $k = d, \ldots, 2$ **do** ▷ Second Half-Sweep

Solve $v^k := \arg\min_v \mathcal{J}(P_k v)$ via (37) using $\hat{S}_k$ and $\hat{B}_k$.

Compute $[Q, R] = \mathrm{qrd}(\mathrm{unf}_R(\mathrm{res}(v^k)))$.

Set $W^k := \mathrm{res}(Q), W^{k-1} := \mathrm{res}\left(\mathrm{unf}_L\left(W^{k-1}\right) \cdot R^\top\right)$.

Update right stack $\mathcal{R}^{k-1}$ using equation (7).

end for

Compute objective $\xi := \mathcal{J}(W^1, \ldots, W^d)$

Set $\epsilon := |\tilde{\xi} - \xi|$ and $\tilde{\xi} := \xi$.

end while

Return Tensor train approximation $\mathcal{T} = \mathcal{A}\left(m, r, \{x^n, y^n\}_{n=1}^N\right) = \left(W^1, \ldots, W^d\right)$.

The map η_j is piecewise smooth and exhibits kinks along the locus where (42) switches between slack and binding.

Endogenous Grid Method. As in Section 3, we reduce the $(M+1)$-dimensional nonlinear system to a single scalar equation by exploiting EGM. The irreversibility constraint is handled by a simple case distinction. For each country j, fix next-period capital k'_j exogenously. Compute the expectations on the right-hand side of (41) via quasi-analytical integration $e_j \equiv \mathbb{E}_t\{\eta_j(x')\}$. First, compute the interior solution candidate. Set $\mu_j = 0$ and solve (41) for the implied current capital $\tilde{k}_j$. Then, check consistency between the assumed Lagrange multiplier and the implied current capital. If $k'_j \geq (1-\delta)\tilde{k}_j$, accept the interior solution $(\tilde{k}_j, \mu_j = 0)$. Otherwise, the constraint binds. Set $\tilde{k}_j = k'_j/(1-\delta)$, $\mu_j = \lambda[1+\phi g_j(\tilde{k}_j, k'_j)] - \beta e_j \geq 0$, which follows by rearranging (41). Given the set $\{\tilde{k}_j\}_{j=1}^M$ implied by the fixed $\{k'_j\}$, substitute into the aggregate budget constraint (17) and solve the resulting scalar equation for the multiplier λ. This completes one EGM update. The method retains all advantages of Section 3 while accommodating the kink induced by (40).

Error Measure. We follow Section 3 and report unit-free Euler errors. With irreversibility, we additionally penalize violations of of (40). Define the percentage shortfall from the lower bound

$$IC^j \equiv 1 - \frac{k'_j}{k_j(1-\delta)}. \tag{44}$$

Let EE^j denote the standard (unit-free) Euler error as in Section 3, accounting for the amendments in (41), evaluated at points where the constraint is slack. We summarize the accuracy in the Euler equation for country j by

$$\max\left\{EE^j, IC^j, \min\left\{-EE^j, -IC^j\right\}\right\}.$$

By construction the error is non-negative, it coincides with $|EE^j|$ when the constraint is non-binding, and reduces to the (percentage) irreversibility violation when it binds. See Brumm and Scheidegger (2017) for a more detailed discussion.

C Details Heterogeneous Agent Model

This appendix expands parts of the main text that were kept brief for space.

C.1 Local Augmented Lagrangian Problem

To apply Algorithm 3 efficiently to the augmented Lagrangian while working in TT format, we must never assemble the global functional $\mathcal{J}$ on the full tensor A nor form the retraction operator P_k. Section 2 showed how left/right contraction stacks — $\mathcal{L}$ and $\mathcal{R}$ — yield a reduced linear system that acts directly on the k-the core W^k. For the master equation in Section 4 we follow the same strategy, but we build separate stacks for each instance of the value function $V, V^{\text{tax}}, V^{\partial_a}, V^{\partial_z}, V^{\partial_{z^2}}, V^{\partial_\Gamma}$. From these we assemble the local projections M_k (interior operator) and C_k (boundary operator) at core k.

Prerequisites and Notation. Index coordinates by $d \in \{a, z, m_1, \dots, m_D\}$, with polynomial basis size m_k along coordinates k. Let $\mathcal{S}$ be the sample set, $|\mathcal{S}| = N$. Precompute evaluation matrices

$$\mathcal{X}_k^{\square,ij} \in \mathbb{R}^{N \times m_k}, \quad \square \in \{V, \partial_a, \partial_z, \partial_{z^2}\partial_\Gamma, \text{tax}\}, \quad i, j \in \{1, 2\},$$

where $\mathcal{X}_k^{\square,ij}$ stores basis values and $\mathcal{X}_k^{\partial_a,ij}, \mathcal{X}_k^{\partial_z,ij}, \mathcal{X}_k^{\partial_{z^2},ij}$ store the corresponding differential operators applied in the basis. The matrix $\mathcal{X}_k^{\text{tax},ij}$ evaluates basis functions at post-tax variables $(\tilde{a}, \tilde{\Gamma})$. Represent V in TT form with cores $W^1, \dots, W^d$ and TT ranks r_k (with $r_0 = r_d = 1$). For each label $\square$ define local left/right stacks by contracting all cores except W^k against the appropriate $\mathcal{X}_\ell^{\square,ij}$ for $\ell \neq k$

$$\mathcal{L}_k^{\square,ij} \in \mathbb{R}^{N \times r_{k-1}}, \quad \mathcal{R}_k^{\square,ij} \in \mathbb{R}^{r_k \times N}.$$

Form the local basis tensor by inserting the k-th dimension

$$\mathcal{B}_{no\ell p}^{\square} = \left(\left(\mathcal{L}_k^{\square,ij} \right)_{no} \cdot \left(\mathcal{X}_k^{\square,ij} \right)_{n\ell} \cdot \left(\mathcal{R}_k^{\square,ij} \right)_{pn} \right)_{no\ell p} \in \mathbb{R}^{N \times r_{k-1} \times m_k \times r_k}. \tag{45}$$

Finally, stack indices (o, ℓ, p) into columns to obtain the local design matrices

$$\boldsymbol{B}^{\square,ij} \in \mathbb{R}^{N \times K_k}, \quad K_k := r_{k-1} m_k r_k,$$

and identify the optimization variable with the vectorized core $v_k := \text{vec}(W^k) \in \mathbb{R}^{K_k}$.

Local Augmented Lagrangian. Given interior and boundary projections $M_k \in \mathbb{R}^{N \times K_k}$ and $C_k \in \mathbb{R}^{M \times K_k}$, the localized subproblem at core k reads

$$\min_{v_k} \frac{1}{2} ||M_k v_k - u||_2^2 + \frac{\gamma}{2} ||C_k v_k - h - t||_2^2 + \mu^\top (C_k v_k - h - t), \tag{46}$$

where $u \in \mathbb{R}^N$ is the (value-independent) interior right-hand side, $h \in \mathbb{R}^M$ is the boundary target, $t \in \mathbb{R}^M$ are non-negativity slacks, $\mu \in \mathbb{R}^M$ are the current multipliers, and $\gamma > 0$ is the penalty.

Interior Operator. The interior projection is the full operator M with the global basis X replaced by the local basis $\boldsymbol{B}$.

$$M_k^{ij} = \left(\rho_i + \frac{1}{\Delta} + \lambda_j + \omega_i + \theta\right) \boldsymbol{B}^{V,ij} - \left(w(z,\Gamma)\varepsilon_j + r(z,\Gamma)a - c_{ij}\right) \boldsymbol{B}^{\partial_a,ij}$$
$$- \kappa(\bar{z} - z)\boldsymbol{B}^{\partial_z,ij} - \frac{1}{2}\sigma_z^2 \boldsymbol{B}^{\partial_{z^2},ij} - \mu(\Gamma)\boldsymbol{B}^{\partial_\Gamma,ij} - \lambda_j \boldsymbol{B}^{V,ij} - \omega_i \boldsymbol{B}^{V,ij} - \theta \boldsymbol{B}^{\text{tax},ij}.$$

Boundary Operator. For the no-borrowing boundary condition we have

$$C_k^{ij} = -\boldsymbol{B}^{\partial_a,ij}, \quad \text{and } h \text{ as in the global problem.}$$

Reduced Normal Equations. Consistent with the general formulation in Appendix A define the stacked system

$$\hat{S}_k = \begin{bmatrix} M_k \\ \sqrt{\gamma} C_k \end{bmatrix}, \quad \hat{B}_k = C_k, \quad \hat{u}_k = \begin{bmatrix} u \\ \sqrt{\gamma}(h + t + \mu/\gamma) \end{bmatrix}.$$

To solve the minimization problem apply Algorithm 3.

C.2 Heterogeneous Agent Solution Algorithm

Algorithm 4 provides a detailed description of the solution method described in Section 4.

Algorithm 4 Solving the Master Equation using TTA

Require: Asset grid $\mathring{a} \in \mathbb{R}^n$; initial distributions $\mathcal{G} = \{g_i\}_{i=1}^N \subset \mathbb{R}^{4n}$, where each g_i stacks the four idiosyncratic-state histogram over $\mathring{a}$; moment generators $\{f_k\}_{k=1}^M$, with $f_k \in \mathbb{R}^{4n}$. For each i, define the moment vector $m_i \in \mathbb{R}^M$ by $[m_i]_k = \langle g_i, f_k \rangle$. Let $\mathcal{S} = \{m_i\}_{i=1}^N \subset \mathbb{R}^M$; moment law of motion $\mu(\Gamma) : (z, m) \mapsto \dot{m}$, mapping today's (z, m) to drifts; tax-event moment forecast $\Lambda(\Gamma) : m \mapsto \tilde{m}$, mapping pre-shock into after-shock moments; state sample $X = \{x_i\}_{i=1}^N \subset \mathbb{R}^{M+2}$ with $x_i = (a_i, z_i, m_i)$ and $\{a_i\}_{i=1}^N \subset \mathbb{R}$; initial value function (TT) for each idiosyncratic state an initial guess $V^{ij}(\cdot; \mathcal{T}_{ij})$; tolerance thresholds $\varepsilon_{\text{ADMM}}$, ε_{VFI}, ε_{FC}, $\varepsilon_{\text{Sim}} > 0$ aggregate shocks $\{z_t\}_{t=1}^T$ and tax indicators $\{\mathbb{1}_t^{\text{tax}}\}_{t=1}^T$.

Simulation, VFI & ADMM

Evaluate $\tilde{v}_{ij} := V^{ij}(X)$.

Initialize ADMM penalty $\gamma > 0$, multiplier μ and slacks $t \geq 0$.

Set residuals $\xi_{\text{ADMM}}, \xi_{\text{VFI}}, \xi_{\text{FC}}, \xi_{\text{Sim}} := \infty$.

while $\xi_{\text{Sim}} > \varepsilon_{\text{Sim}}$ **do** ▷ Simulation Loop

  while $\xi_{FC} > \varepsilon_{\text{FC}}$ **do** ▷ Forecast Loop

    Set $v_{ij} := \tilde{v}_{ij}$.

    while $\xi_{\text{VFI}} > \varepsilon_{\text{VFI}}$ **do** ▷ VFI Loop

      compute $c_{ij}^* = (u')^{-1}\left(\partial_a V^{ij}(X; \mathcal{T}_{ij})\right)$ from FOC (29).

      while $\xi_{\text{ADMM}} > \varepsilon_{\text{ADMM}}$ **do** ▷ ADMM Loop

        Minimize the augmented Lagrangian (33) using ALS (3).

        Update slack and multiplier using equations (34) and (35).

        Adjust the penalty γ to balance primal and dual residuals.

        Set $\xi_{\text{ADMM}} := ||t_{\text{new}} - t_{\text{old}}||_2^2$

      end while

      Update $V^{ij} = V(\cdot; \mathcal{T}_{ij})$ and evaluate $v'_{ij} := V^{ij}(X)$.

      Set $\xi_{\text{VFI}} := ||v'_{ij} - v_{ij}||$ and $v_{ij} := v'_{ij}$.

    end while

    On $\mathcal{G}$ compute $\{\dot{g}_i\}_{i=1}^N$ using $V^{ij}(\cdot; \mathcal{T}_{ij})$, then compute $\{\dot{m}_i\}_{i=1}^N$.

    Fit the updated law of motion $\tilde{\mu}(\Gamma) : (z, m) \mapsto \dot{m}$, and set $\mu(\Gamma) := \tilde{\mu}(\Gamma)$

    Set $\xi_{\text{FC}} = ||v'_{ij} - \tilde{v}_{ij}||$, $\tilde{v}_{ij} := v'_{ij}$.

  end while

  Simulate $\tilde{\mathcal{G}} := \{\tilde{g}_i\}_{i=1}^N$ on the grid $\mathring{a}$ using $V^{ij}(\cdot; \mathcal{T}_{ij})$, starting at g_1.

  Select moments $\{f_k\}_{k=1}^M$ using a posteriori model reduction (4.2).

  Fit updated tax-shock forecast $\tilde{\Lambda}(\Gamma) := m \mapsto \tilde{m}$, and set $\Lambda(\Gamma) := \tilde{\Lambda}(\Gamma)$.

  Set $\xi_{\text{Sim}} := ||\langle \tilde{g}, f \rangle - \langle g, f \rangle||$, and $\mathcal{G} := \tilde{\mathcal{G}}$.

end while

Compute error measure.

References

Achdou, Y., Han, J., Lasry, J.-M., Lions, P.-L., & Moll, B. (2021). Income and wealth distribution in macroeconomics: A continuous-time approach. *The Review of Economic Studies*, *89*(1), 45–86.

Ahn, S., Kaplan, G., Moll, B., Winberry, T., & Wolf, C. (2018). When inequality matters for macro and macro matters for inequality. *NBER macroeconomics annual*, *32*(1), 1–75.

Auclert, A., Bardóczy, B., Rognlie, M., & Straub, L. (2021). Using the sequence-space jacobian to solve and estimate heterogeneous-agent models. *Econometrica*, *89*(5), 2375–2408.

Auclert, A., Rognlie, M., & Straub, L. (2025). Fiscal and monetary policy with heterogeneous agents. *Annual Review of Economics*, *17*(Volume 17, 2025), 539–562.

Azinovic, M., Gaegauf, L., & Scheidegger, S. (2022). Deep equilibrium nets. *International Economic Review*, *63*(4), 1471–1525.

Azinovic, M., & Zemlicka, J. (2024). Intergenerational consequences of rare disasters.

Bachmayr, M. (2023). Low-rank tensor methods for partial differential equations. *Acta Numerica*, *32*, 1–121.

Bayer, C., & Luetticke, R. (2020). Solving discrete time heterogeneous agent models with aggregate risk and many idiosyncratic states by perturbation. *Quantitative Economics*, *11*(4), 1253–1288.

Bayer, C., Luetticke, R., Weiss, M., & Winkelmann, Y. (2025). An endogenous gridpoint method for distributional dynamics [Forthcoming]. *Journal of Monetary Economics*.

Bhandari, A., Bourany, T., Evans, D., & Golosov, M. (2023, September). A perturbational approach for approximating heterogeneous agent models (Working Paper No. 31744). National Bureau of Economic Research.

Bigoni, D., Engsig-Karup, A. P., & Marzouk, Y. M. (2016). Spectral tensor-train decomposition. *SIAM Journal on Scientific Computing*, *38*(4), A2405–A2439.

Bilal, A. (2023, April). Solving heterogeneous agent models with the master equation (Working Paper No. 31103). National Bureau of Economic Research.

Brumm, J., Krause, C., Schaab, A., & Scheidegger, S. (2022). Sparse grids for dynamic economic models. In *Oxford research encyclopedia of economics and finance*.

Brumm, J., & Scheidegger, S. (2017). Using adaptive sparse grids to solve high-dimensional dynamic models. *Econometrica*, *85*(5), 1575–1612.

Carroll, C. D. (2006). The method of endogenous gridpoints for solving dynamic stochastic optimization problems. *Economics Letters*, *91*(3), 312–320.

Christensen, B. J., Neri, L., & Parra-Alvarez, J. C. (2024). Estimation of continuous-time linear dsge models from discrete-time measurements. *Journal of Econometrics*, *244*(2), 105871.

Dektor, A., Rodgers, A., & Venturi, D. (2021). Rank-adaptive tensor methods for high-dimensional nonlinear pdes. *Journal of Scientific Computing*, *88*(2), 36.

Dolgov, S., Kalise, D., & Kunisch, K. K. (2021). Tensor decomposition methods for high-dimensional hamilton–jacobi–bellman equations. *SIAM Journal on Scientific Computing*, *43*(3), A1625–A1650.

Duarte, V., Duarte, D., & Silva, D. H. (2024). Machine learning for continuous-time finance. *The Review of Financial Studies*, *37*(11), 3217–3271.

Fernández-Villaverde, J., Hurtado, S., & Nuño, G. (2023). Financial frictions and the wealth distribution. *Econometrica*, *91*(3), 869–901.

Fernández-Villaverde, J., Nuno, G., Sorg-Langhans, G., & Vogler, M. (2020). Solving high-dimensional dynamic programming problems using deep learning. *Unpublished working paper*.

Fernández-Villaverde, J., Nuño, G., & Perla, J. (2024). Taming the curse of dimensionality: Quantitative economics with deep learning (tech. rep.). National Bureau of Economic Research.

Folini, D., Friedl, A., Kübler, F., & Scheidegger, S. (2025). The climate in climate economics. *Review of Economic Studies*, *92*(1), 299–338.

Gopalakrishna, G. (2021). Aliens and continuous time economies. *Swiss Finance Institute Research Paper*, (21-34).

Gorodetsky, A., Karaman, S., & Marzouk, Y. (2019). A continuous analogue of the tensor-train decomposition. *Computer Methods in Applied Mechanics and Engineering*, *347*, 59–84.

Gorodnichenko, Y., Maliar, L., Maliar, S., & Naubert, C. (2021). Household savings and monetary policy under individual and aggregate stochastic volatility.

Grasedyck, L., & Krämer, S. (2019). Stable als approximation in the tt-format for rank-adaptive tensor completion. *Numerische Mathematik*, *143*(4), 855–904.

Gu, Z., Lauriere, M., Merkel, S., & Payne, J. (2024). Global solutions to master equations for continuous time heterogeneous agent macroeconomic models. *arXiv preprint arXiv:2406.13726*.

Han, J., Yang, Y., & E, W. (2021). Deepham: A global solution method for heterogeneous agent models with aggregate shocks. *arXiv preprint arXiv:2112.14377*.

Holtz, S., Rohwedder, T., & Schneider, R. (2012). The alternating linear scheme for tensor optimization in the tensor train format. *SIAM Journal on Scientific Computing*, *34*(2), A683–A713.

Huang, J. (2023). Breaking the curse of dimensionality in heterogeneous-agent models: A deep learning-based probabilistic approach. *Available at SSRN 4649043*.

Judd, K. L., Maliar, L., Maliar, S., & Valero, R. (2014). Smolyak method for solving dynamic economic models: Lagrange interpolation, anisotropic grid and adaptive domain. *Journal of Economic Dynamics and Control*, *44*, 92–123.

Kahou, M. E., Fernández-Villaverde, J., Perla, J., & Sood, A. (2021). Exploiting symmetry in high-dimensional dynamic programming (tech. rep.). National Bureau of Economic Research.

Kase, H., Melosi, L., & Rottner, M. (2022). *Estimating nonlinear heterogeneous agents models with neural networks*. Centre for Economic Policy Research.

Kollmann, R., Maliar, S., Malin, B. A., & Pichler, P. (2011). Comparison of solutions to the multi-country real business cycle model. *Journal of Economic Dynamics and Control*, *35*(2), 186–202.

Krueger, D., & Kubler, F. (2006). Pareto-improving social security reform when financial markets are incomplete!? *American Economic Review*, *96*(3), 737–755.

Krusell, P., & Smith, A. A., Jr. (1998). Income and wealth heterogeneity in the macroeconomy. *Journal of political Economy*, *106*(5), 867–896.

Maliar, L., Maliar, S., & Winant, P. (2021). Deep learning for solving dynamic economic models. *Journal of Monetary Economics*, *122*, 76–101.

Nuño, G., Renner, P., & Scheidegger, S. (2024). Monetary policy with persistent supply shocks.

Oseledets, I. V. (2011). Tensor-train decomposition. *SIAM Journal on Scientific Computing*, *33*(5), 2295–2317.

Payne, J., Rebei, A., & Yang, Y. (2025). Deep learning for search and matching models. *Available at SSRN 5123878*.

Reiter, M. (2009). Solving heterogeneous-agent models by projection and perturbation. *Journal of Economic Dynamics and Control*, *33*(3), 649–665.

Richter, L., Sallandt, L., & Nüsken, N. (2021, 18–24 Jul). Solving high-dimensional parabolic pdes using the tensor train format. In M. Meila & T. Zhang (Eds.), *Proceedings of the 38th international conference on machine learning* (pp. 8998–9009, Vol. 139). PMLR.

Richter, L., Sallandt, L., & Nüsken, N. (2024). From continuous-time formulations to discretization schemes: Tensor trains and robust regression for bsdes and parabolic pdes. *Journal of Machine Learning Research*, *25*(248), 1–40.

Sauzet, M. (2021). Projection methods via neural networks for continuous-time models. *Available at SSRN 3981838*.

Schaab, A. (2020). Micro and macro uncertainty.

Schaab, A., & Zhang, A. (2022). Dynamic programming in continuous time with adaptive sparse grids. *Available at SSRN 4125702*.