SOLVING HAMILTON-JACOBI BELLMAN EQUATIONS ON ADAPTIVE SPARSE GRIDS

Jochen Garcke
joint work with Ilja Kalmykov, Axel Kröner, Steffen Rutscheidt

UNIVERSITÄT BONN
Institute for Numerical Simulation

Fraunhofer SCAI
Department Numerical Data-Driven Prediction

Fraunhofer Center for Machine Learning
Application-oriented research for immediate benefit to the economy and to the benefit of society

- **72 institutes** and research units

- **26,600 staff**

- **€2.6 billion annual research budget** totaling. Of this sum, more than €2.2 billion is generated through contract research
  - Roughly two thirds of this sum is generated through contract research on behalf of industry and publicly funded research projects
  - Roughly one third is contributed by the German federal and countries governments in the form of base funding
Outline

1. Sparse Grids
2. Semi-Lagrangian Scheme for Optimal control
3. Higher Order Runge-Kutta Methods
4. Finite Differences on Sparse Grids for Economic Problems
**Hierarchical Basis**

(a) hierarchical basis $V_3 = W_3 \oplus W_2 \oplus V_1$

(b) fold-out hierarchical basis

(c) interpolation

(d) last basis before the boundary is folded up and extrapolated linearly
Hierarchical Basis Functions in Higher Dimensions

- \( d \)-dimensional piecewise \( d \)-linear functions
  \[
  \phi_{l,j}(x) := \prod_{t=1}^{d} \phi_{l,t,j_t}(x_t)
  \]

- Hierarchical difference space \( \mathcal{W}_l \) (\( \varepsilon_t \) is \( t \)-th unit vector)
  \[
  \mathcal{W}_l := \mathcal{V}_l \setminus \bigcup_{t=1}^{d} \mathcal{V}_l - \varepsilon_t,
  \]

- Hier. diff. space represented by \( \mathcal{W}_l = \text{span}\{\phi_{l,j} | j \in B_l\} \) with
  \[
  B_l := \left\{ j \in \mathbb{N}^d \mid \begin{array}{l}
  j_t = 1, \ldots, 2^{l_t} - 1, \ j_t \text{ odd,} \\
  j_t = 0, 1, 2, \quad t = 1, \ldots, d, \text{ if } l_t > 1, \\
  j_t = 0, 1, 2, \quad t = 1, \ldots, d, \text{ if } l_t = 1
  \end{array} \right\}.
  \]

- Full grid space in hierarchical basis
  \[
  \mathcal{V}_n^s := \bigoplus_{|l|_\infty \leq n} \mathcal{W}_l
  \]
Hierarchical Subspaces $W_i$
Sparse Grids

- we define the sparse grid function space $V^s_n \subset V_n$ as

\[ V^s_n := \bigoplus_{|\ell_1| \leq n+d-1} W_{\ell_1} \]

- every $f \in V^s_n$ can now be represented as

\[ f_n^s(x) = \sum_{|\ell_1| \leq n+d-1} \sum_{j \in B_{\ell_1}} \alpha_{\ell_1,j} \phi_{\ell_1,j}(x) \]

- approximation property in $H^2_{mix}$

\[ \| f - f_n^s \|_2 = O(h_n^2 \log(h_n^{-1})^{d-1}) \]

- sparse grid needs $O(h_n^{-1}(\log(h_n^{-1}))^{d-1})$ points
Sparse Tensor Decomposition

- Sparse grids introduced by Zenger, 1991
- Original idea can be identified in work of Smolyak, 1963
- Other related approaches using construction exists as well
- Overview from Bungartz and Griebel in Acta Numerica 2004
- Tutorial introduction in G. 2013
- Truncated tensor construction can be used for "any" multi-resolution representation
  - Polynomials
  - $B$-splines
  - Wavelets
  - Fourier series (hyperbolic cross approximation)
  - Multi-level Monte Carlo (different resolutions for random variables as well as discretizations)
  - Time vs. space discretization
- Sparsity pattern can be generalized from simplex or can be dimension-adaptive
- SG++ library from Dirk Pflüger (U. Stuttgart) with Matlab and Python-bindings
- In UQ-context and for numerical integration further libraries exist
Problems with Sparse Grids for HJB: Monotonicity

- Interpolation of peaked Gaussian function with sparse grid of level \( n = 2 \)
  \[
  f(x_1, x_2) := \exp(-100(x_1 - 0.5)^2) \times \exp(-100(x_2 - 0.5)^2)
  \]

- Sparse grid interpolation does not preserve positivity
Problems with Sparse Grids for HJB: Monotonicity

- Problem persists for monotonically increasing (strictly) concave functions, e.g.
  \[ f(x, y) = \frac{-1}{1 + 10x + 10y} + 50. \]
- Function is similar to value functions arising for heterogenous agent models in economics.

function plot of \( f \)

interpolation plot of \( f \)
Spatially Adaptive Sparse Grids

- Spatially adaptive grids can be used when functions
  - do not fulfill smoothness condition or
  - strongly vary due locally large derivatives
- Adaptivity helps to some degree to cope with non-monotonicity
  - Refine one grid point by creating all children (middle)
  - To keep grid consistent, missing parents are created (right)

Usually hierarchical surplus $\alpha_{l,j}$ is used as refinement indicator

In case of monotonically increasing concave functions one could check for negative derivatives
Optimal Feedback Control

- optimal feedback control of a dynamical system

$$\begin{aligned}
\min_{u \in \mathcal{U}_{ad}} & \quad J(u) = \int_0^T l(y(t), u(t)) \, dt, \\
\text{s.t.} & \quad \dot{y}(t) = f(y(t), u(t)), \quad t > 0 \\
& \quad y(0) = y_0
\end{aligned}$$

- **STATE** $y(t) \in \mathbb{R}^d$, initial state $y_0 \in \mathbb{R}^d$
- **CONTROL** $u(t) \in U^m \subset \mathbb{R}^m$ (often called action)
- Lipschitz continuous **DYNAMICS** $f : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}^d$
- running **COST** with polynomial growth $l : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}$
- set of **ADMISSIBLE CONTROLS**
  $$\mathcal{U}_{ad} = \{ u \in L^2([0, T]; U^m) \mid U^m \subset \mathbb{R}^m \text{ compact} \}$$
- aim: **FEEDBACK LAW**
  $$u^* = K(t, y^*(t))$$
Basic Sparse Grid Semi-Lagrangian Scheme

evaluate for $x \in Q_I$, $\Delta t > 0$, $K = T/\Delta t$, $k = K - 1, \ldots, 0$,

\[
\begin{cases}
    v^k(x) = \min_{u \in U} \left( \Delta t l(x, u) + v^{k+1}(y_x(\Delta t)) \right), \\
    v^k(x) = 0
\end{cases}
\]

$y_x(\Delta t)$ state obtained by time discretization scheme from $x$
minimization either by comparing $v^{k+1}$ values over finite set, nonlinear optimization, or gradient $\nabla v$

Algorithm 1: Adaptive SL-SG scheme

Data: refinement constant $\varepsilon$, coarsening constant $\eta$
Result: sequence of adaptive sparse grid solutions $v_k \in V_{I(k)}$
initialize $I(K)$
for $k = K - 1, \ldots, 0$ do
    initialize $I(k - 1)$ with $I(k)$
    adaptively interpolate $\min_{u \in U} \left( v_k(y_x(\Delta t)) + \Delta t l(x, u) \right)$
    coarsen $v_{k-1} \in V_{I(k-1)}$
iterate in time with $\Delta t = T/K$
compute $v_{k-1}$

see Bokanowski, G., Griebel, and Klomppmaker (2013)
**Example 2D: Simplified Semi-Discrete Wave Equation**

- semi-discrete PDE control problem (following Kröner, Kunisch, Zidani (2015))
- formulate as **FIRST ORDER SYSTEM** in time with $y^1 = y$, $y^2 = \dot{y}$
- dynamics $f(x, u) = Ax + Bu$
- with running cost $l(x, u) := \beta_x x_1^T M x_1 + \beta_u u^T u$
- **Hamilton-Jacobi Bellman equation** with dimension $2d$
- first consider a simplified example based on harmonic oscillator

\[
\beta_x = 2, \quad \beta_u = 0.1, \quad T = 1, \quad \Delta t = 0.01,
\]

\[
A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 1 \end{pmatrix},
\]

initial data $x \in \mathbb{R}^2$, domain $Q = [-1, 1]^2$, $U = [-3.5, 3.5]$.
- reference solution computed with a higher order FD code on very fine mesh
- further details in G., Kröner (2017)
**Example 2D: Convergence of value function**

(e) $L_2$ error in the value function vs. $\varepsilon$

(f) $L_2$ error in the value function vs. nodes
Example 2D: Value function

Figure: adaptive sparse grid with normal and fold out hat functions

(g) $\varepsilon = 1.95_{-4}$

(h) $\varepsilon = 1.95_{-4}$
Example 2D: Convergence in the Trajectory

(a) $L_{\infty}$ error in the trajectory vs. $\varepsilon$

(b) $L_{\infty}$ error in the trajectory vs. nodes
**Example 2D: Convergence in the Control**

- **Figure (c)**: $L_\infty$ error in the control vs. $\varepsilon$.

- **Figure (d)**: $L_\infty$ error in the control vs. nodes.
Example: Semi-Discrete Wave Equation (4D)

- Now wave equation as first order system in time
  \[
  \dot{y}(t) = f_w(y(t), u(t)), \quad t > 0, \quad y(0) = y_0
  \]

- With dynamics
  \[
  f_w : \mathbb{R}^{2d} \times \mathbb{R}^m \to \mathbb{R}, \quad f_w(x, u) := Ax + Bu
  \]
  where
  \[
  A := \begin{pmatrix} 0 & I_d \\ -cM^{-1}A & 0 \end{pmatrix}, \quad B := \begin{pmatrix} 0 \\ b \end{pmatrix}, \quad b \in \mathbb{R}^{m \times d}, \quad y_0 \in \mathbb{R}^{2d}
  \]

- Cost
  \[
  l_w(x, u) := \beta_x x_1^T M x_1 + \beta_u u^T u
  \]

- Consider setup
  \[
  \beta_x = 2, \quad \beta_u = 0.1, \quad T = 4, \quad \Delta t = 0.01, \quad c = 0.05.
  \]

- Compute reference trajectories in state space and control space using a Riccati approach
EXAMPLE: CONVERGENCE IN THE CONTROL (4D)

(e) $L_\infty$ error in the control vs. $\varepsilon$

(f) $L_\infty$ error in the control vs. nodes
Example: semi-discrete wave equation (4D)

(g) $y_1$

(h) $y_3$

(i) $u_1$

(j) $y_2$

(k) $y_4$

(l) $u_2$
**Example: Semi-discrete Wave Equation (6D)**

(a) Error in the control vs. $\varepsilon$

(b) Error in the control vs. nodes

Need to decrease time step to $\Delta t = 0.0025$ (larger entries in stiffness matrix)
EXAMPLE: SEMI-DISCRETE WAVE EQUATION (8D)

(c) error in the control vs. \( \varepsilon \)

(d) error in the control vs. nodes

need to decrease time step to \( \Delta t = 0.00125 \) (larger entries in stiffness matrix)
 Higher Order Runge-Kutta Methods

- look at control formulation

\[
\begin{align*}
\min_{u \in U_{ad}} J(u) &= \int_{0}^{T} l(y(t), u(t)) \, dt, \quad \text{s.t.} \\
\dot{y}(t) &= f(y(t), u(t)), \quad t > 0, y(0) = y_0
\end{align*}
\]

- we need time discretisation for \( y(t) \) and
- quadrature rule to evaluate \( \min_{u \in U_{ad}} J(u) \)
- with higher order RK-scheme this could look like (misusing notation)

\[
v^k(x) = \min_{u_1, u_2, u_3, \ldots} \left( \Delta t \sum_i c_i l(x^{\tau_i}, u_i) + v^{k+1}(\hat{y}_x(\Delta t)) \right)
\]

where \( \hat{y}_x \) takes into account actions \( u_i \) and RK-scheme with intermediate steps \( \tau_i \)
- therefore RK4 would have \( O(d_c^4) \) complexity if complexity for computing one control is \( d_c \)
- see also Falcone, Ferretti (1994)
**Structure Preserving Runge-Kutta Methods**

- observe that for explicit or diagonally implicit RK schemes, the last element $u^n_s$ of discrete control vector affects only the last RK-stage $k_s$
- formulate condition on RK-coeff., s.t. control optimization is applied separately to each stage
- idea is to mimic Dynamic Programming property within single step of semi-Lagrangian scheme
- class of RK methods fulfilling this: diagonally implicit symplectic Runge-Kutta (DISRK) schemes
- originally developed for long time integration of Hamiltonian systems
- “re-use” implicit collocation RK-scheme for quadrature
- DISRK are equivalent to composition of implicit midpoint schemes $\Psi_h = \Phi_{\gamma_s \tau} \circ \cdots \circ \Phi_{\gamma_2 \tau} \circ \Phi_{\gamma_1 \tau}$
- we can construct a SL scheme, which has $O(d_c \cdot s)$ complexity for the minimization problem
- for example DISRK$_5$ is a method of order 4 using 5 steps

$$
\gamma_1 = \gamma_2 = \gamma_4 = \gamma_5 = \frac{1}{4 - 4^{1/(p+1)}}, \quad \gamma_3 = -\frac{4^{1/(p+1)}}{4 - 4^{1/(p+1)}} \tag{1}
$$

- observe for $s > 1$ DISRK goes backwards in time for some steps !
- details in G., Kalmykov (2018)
Harmonic Oszillator as Simplified Semi-Discrete Wave - Revisited

- consider again the simplified example based on harmonic oscillator

$$\beta_x = 2, \beta_u = 0.1, \quad T = 0.1, \Delta t = 0.01,$$

$$A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

initial data $x \in \mathbb{R}^2$, domain $Q = [0,1]^2$, $U = [-3.5, 3.5]$.

- control space discretization is denoted by $A_\Delta$

- use sparse grids with linear and 3rd order B-Splines, both with fold out ansatz (SG++-library)

- equation for stage $k$ of Implicit Mipoint Rule is solved with standard Newton algorithm

- obtain a reference solution from continuous time Riccati differential equation

- computation of error estimates is performed on a domain $\Omega_{\text{ref}} = [0.25, 0.75]^2$
Simplified Semi-Discrete Wave - Revisited with DISRK1

(e) implicit midpoint, lin. sparse grids

(f) implicit midpoint, sparse grids 3rd order B-splines
Simplified Semi-Discrete Wave

\[
\begin{align*}
10^{-21} & \quad 10^{-13} \\
10^{-12} & \quad 10^{-11} \\
10^{-10} & \quad 10^{-9} \\
10^{-8} & \quad 10^{-7} \\
10^{-6} & \quad 10^{-5} \\
10^{-4} & \quad 10^{-3} \\
10^{-2} & \quad 10^{-1} \\
3.85 & \quad 5.6 \\
\end{align*}
\]

\(\tau\) vs. \(e_2^v\)

DISRK\(_5\) vs. DISRK\(_7\)

- 8, \(|A_\Delta| = 5 \cdot 10^4\)
- 8, \(|A_\Delta| = 5 \cdot 10^6\)
- 8, \(|A_\Delta| = 1 \cdot 10^5\)
- 9, \(|A_\Delta| = 5 \cdot 10^6\)
finite differences on sparse grids were introduced by [Griebel.Schiekofer:1998]

there, finite difference operators are a composition of three partial operators

1. a basis transformation from nodal to hierarchical basis in all dimensions but the dimension $j$ in which we aim to use the finite difference stencil

2. application of a finite difference stencil in dimension $j$, where mesh size is given as local step size to the neighboring grid point in dimension $j$

3. a basis transformation from hierarchical to nodal basis in all dimensions but dimension $j$

consistency proofs can be given for elliptic PDE model problems

we investigate an alternative from [Ahn.2017], where additional ghost nodes are used

instead of using function values on the sparse grid points, one interpolates on ghost nodes

therefore, one does not need to use specific basis transformations and one can simply take any sparse grid library, such as SG++ [Pflüger:2010]
Finite Differences on Sparse Grids Using Ghost Nodes

- we define the ghost node step size $h_{g_j}$ in dimension $j$, $1 \leq j \leq n$, for a grid point $x_{l,i}$ by $h_{g_j} := 2^{-k_j}$ where $k_j$ denotes the maximal level used in dimension $j$.

- for a grid point $x_{l,i}$ in which we aim to compute the finite differences in dimension $j$, $1 \leq j \leq d$, we define the corresponding forward difference ghost node by

  $$g_{l,i}^{F,j} := (x_{l,1}, i_1, \ldots, x_{l,j}, i_j + h_{g_j}, \ldots, x_{l,d}, i_d)$$

- for backward difference corresponding backward difference ghost nodes are defined accordingly.

- left: ghost node (red) that is used for sparse grid forward FD in $x$-dimension in green grid point.

- right: all forward difference ghost nodes that are used for the sparse grid are drawn in red.
**Finite Differences on Sparse Grids Using Ghost Nodes**

- with interpolation operator on the sparse grid

\[ I_s : \bigoplus_{k=1}^{l} W_k \rightarrow V_l. \]

- and interpolation operator for the by \( h_{gj} \) shifted sparse grid (grid of ghost nodes)

\[ I_{h_{gj}}^F : \bigoplus_{k=1}^{l} W_k \rightarrow V_l \quad \text{and} \quad I_{h_{gj}}^B : \bigoplus_{k=1}^{l} W_k \rightarrow V_l. \]

where \( I_{h_{gj}}^F \) is the forward difference and \( I_{h_{gj}}^B \) the backward difference in dimension \( j \), \( 1 \leq j \leq d \)

- we can define the *sparse grid backward difference operator* \( \tilde{D}_{j}^{S,B} \) by

\[ \tilde{D}_{j}^{S,B} := I_s - I_{h_{gj}}^B : \bigoplus_{k=1}^{l} W_k \rightarrow V_l. \]

- for sparse grids in 2D up to level 3 we confirmed the equivalence to [Griebel.Schiekofer:1998]

- an extended version of [Ahn:2017] will cover the relation of the two FD approaches in more detail
ECONOMIC MODEL PROBLEM FROM [KAPLAN.MOLL.VIOLANTE:2019]

want to solve the maximization problem

$$\max_{\{c_t,d_t\}_{t \geq 0}} \mathbb{E}_0 \int_0^\infty e^{-\rho t} u(c_t) dt$$

subject to

$$\dot{b}_t = w z_t r^b(b_t) b_t - d_t - \chi(d_t,a_t) - c_t \quad \dot{a}_t = r^a a_t + d_t$$
$$z_t = \text{Poisson with intensities } \lambda(z,z') \quad b_t \geq b, \ a_t \geq 0.$$
2D model problem

- Errors for value function and all policy functions for regular sparse grids of different levels
- Reference solution is computed on a 600 × 600 full grid
**4D model problem**

- $l_2$-error of value functions
- $l_2$-error of deposit policy $a$ functions

- Errors for value function and one deposit policy functions for adaptive sparse grids
- Reference solution is on sparse grid of level 8 (largest computable in reasonable time)
6D model problem

- $l_2$-error of value functions
- $l_2$-error of deposit policy functions

- errors for value function and one deposit policy functions for adaptive sparse grids
- reference solution is on sparse grid of level 6 (largest computable in reasonable time)
CONCLUSION

- use SL-scheme on sparse grids for HJB equations
- investigate model reduction for optimal feedback control PDE problems leading to HJB equations
- suitable for examples in higher dimensions seen up to 8D
- somewhat more complicated function we treated with sparse grids for front propagation models in Bokanowksi, G., Griebel, Klompmaker (2013)
- large scale parallel studies for economic problem in up to 20D by Brumm, Scheidegger (2017)
- higher order time schemes and higher order discretisation can be effective for smooth problems
- sparse grids with B-splines converge "nicer"
  but have higher computing costs due to need to solve a linear equation system
- investigate FD on sparse grids for economic problems using policy iteration
**Key References**


