MAX PLANCK INSTITUTE
FOR DYNAMICS OF COMPLEX
TECHNICAL SYSTEMS
MAGDEBURG

CSC
COMPUTATIONAL METHODS IN
SYSTEMS AND CONTROL THEORY

Comprehensive Memory-Bound Simulations on Single Board Computers

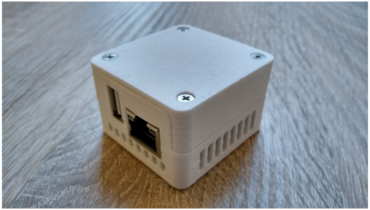
Christian Himpe, Tobias Leibner*, Stephan Rave*

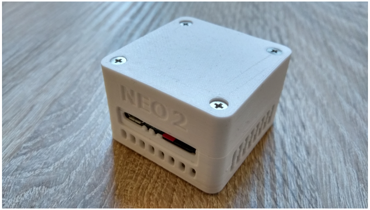
Max Planck Institute Magdeburg
*University of Münster

2nd Workshop on Power Aware Computing (#paco2017)
2017–07–07

Motivation

- Single board computers (SBC) are widely available!
- Increasing performance of SBC (mobile) CPUs!
- SBCs have vector units for number crunching!





Outline

1. (Nonlinear) Model Order Reduction
2. Memory Economic Computation (HAPOD)
3. Numerical Experiments (with SBCs)

Model Order Reduction (MOR)

- Differential Equation System Models
- High-Dimensional State Space
- Fast(er) Solves
- Repeated Solution (Inputs, Parameters)

Special Case: (My) Nonlinear MOR

- Data-Driven
- Empirical Gramians
- No Hyperreduction
- Application: Gas network simulation (MathEnergy)

Input-Output Systems

Nonlinear Ordinary Differential Equation (ODE):

$$\dot{x}(t) = f(t, x(t))$$

Nonlinear Input-Output System:

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t)) \\ y(t) &= g(x(t), u(t)) \end{aligned}$$

Model Reduction

Reduced Order Model (ROM):

$$\dot{x}_r(t) = f_r(x_r(t), u(t))$$
$$y_r(t) = g_r(x_r(t), u(t))$$

ROM Properties:

- $\dim(x_r(t)) \ll \dim(x(t))$
- $\|y - y_r\| \ll 1$

Projection-Based Model Reduction

Projection-Based ROM:

$$\begin{aligned} \dot{x}_r(t) &= V_1 f(U_1 x_r(t), u(t)) \\ y_r(t) &= \quad g(U_1 x_r(t), u(t)) \end{aligned}$$

Truncated Projection Properties:

- $V_1 U_1 = \mathbb{1}$
- $\mathrm{rank}(U_1) = \mathrm{rank}(V_1) \ll \dim(x(t))$

Linear Model Reduction

Linear Input-Output System:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$

Linear ROM:

$$\begin{aligned} \dot{x}_r(t) &= A_r x_r(t) + B_r u(t) \\ y_r(t) &= C_r x_r(t) \end{aligned}$$

Linear Projection-Based ROM:

$$\begin{aligned} \dot{x}_r(t) &= (V_1 A U_1) x_r(t) + (V_1 B) u(t) \\ y_r(t) &= \quad (C U_1) x_r(t) \end{aligned}$$

Balanced Truncation[1]

Controllabiliy Gramian Matrix & Observability Gramian Matrix:

$$W_C := \int_0^\infty \mathrm{e}^{At} \, B B^\intercal \, \mathrm{e}^{A^\intercal} \, \mathrm{d}t, \quad W_O := \int_0^\infty \mathrm{e}^{A^\intercal t} \, C^\intercal C \, \mathrm{e}^{At} \, \mathrm{d}t$$

Balancing and Truncation[2]:

$$W_C = L_C L_C^\intercal, \quad W_O = L_O L_O^\intercal$$

$$L_O L_C^\intercal \overset{\text{SVD}}{=} U D V$$

$$U_1 = U_{*,k}, \quad V_1 = V_{k,*}, \quad \varphi(D, k) < \varepsilon$$

[1] B.C. Moore. **Principal component analysis in linear systems: controllability, observability, and model reduction**. IEEE Transactions on Automatic Control 26(1): 17–32, 1981.

[2] M.S. Tombs and I. Postlethwaite. **Truncated balanced realization of stable, non-minimal state-space systems**. International Journal of Control 46: 1349–1330, 1989.

Cross Gramian[3]

Cross Gramian Matrix:

$$W_X := \int_0^\infty \mathrm{e}^{At}\, BC\, \mathrm{e}^{At}\, \mathrm{d}t$$

Approximate Balancing and Truncation[4]:

$$W_X \overset{\mathsf{SVD}}{=} UDV$$
$$U_1 = U_{*,k}, \quad V_1 = U_1^\intercal, \quad \tilde{\varphi}(D,k) < \varepsilon$$

- $\dim(u(t)) \overset{!}{=} \dim(y(t))$
- $C(\mathbb{1}s - A)^{-1}B = (C(\mathbb{1}s - A)^{-1}B)^\intercal \Rightarrow W_X = \mathsf{BT}$

[3] K.V. Fernando and H. Nicholson. **On the Structure of Balanced and Other Principal Representations of SISO Systems**. IEEE Transactions on Automatic Control 28(2): 228–231, 1983.

[4] D.C. Sorensen and A.C. Antoulas. **Projection methods for balanced model reduction**. Rice University TR01-03: 1–18, 2001.

Linear Cross Gramian[5] [6]:

$$
\begin{aligned}
W_X &= \int_0^\infty \mathrm{e}^{At} BC \, \mathrm{e}^{At} \, \mathrm{d}t \\
&= \int_0^\infty (\mathrm{e}^{At} B)(C \, \mathrm{e}^{At}) \, \mathrm{d}t \\
&= \int_0^\infty (\mathrm{e}^{At} B)(\mathrm{e}^{A^\intercal t} C^\intercal)^\intercal \, \mathrm{d}t
\end{aligned}
$$

- Primal impulse response: $g(t) = \mathrm{e}^{At} B$
- Adjoint impulse response: $\bar{g}(t) = \mathrm{e}^{A^\intercal t} C^\intercal$

[5] K.V. Fernando and H. Nicholson. **On the Cross-Gramian for Symmetric MIMO Systems**. IEEE Transactions on Circuits and Systems 32(5): 487–489, 1985.

[6] H.R. Shaker. **Generalized Cross-Gramian for Linear Systems**. Proceedings of the IEEE Conference on Industrial Electronics and Applications: 749–751, 2012.

Empirical Cross Gramian[7] [8]

Empirical Cross Gramian Matrix:

$$\widehat{W}_X := \sum_{m=1}^{\dim(u(t))} \int_0^\infty \Psi^m(t)\,\mathrm{d}t$$

$$\Psi_{ij}^m(t) = (x_i^m(t) - \bar{x}_i^m)(y_m^j(t) - \bar{y}_m^j)$$

- State trajectories (m-th perturbed input): $x^m(t)$
- Output trajectories (j-th perturbed init state): $y^j(t)$

[7] S. Streif, R. Findeisen and E. Bullinger. **Relating Cross Gramians and Sensitivity Analysis in Systems Biology**. Theory of Networks and Systems 10.4: 437–442, 2006.

[8] C. H. and M. Ohlberger. **Cross-Gramian Based Combined State and Parameter Reduction for Large-Scale Control Systems**. Mathematical Problems in Engineering 2014: 1–13, 2014.

Column-Wise Computation[9]

Column-Wise Representation:

$$\widehat{W}_X = \begin{pmatrix} w_{X,1} & w_{X,2} & \dots & w_{X,N} \end{pmatrix}$$

Empirical Cross Gramian Column:

$$w_{X,j} = \sum_{m=1}^{\dim(u(t))} \int_0^\infty \psi^{jm}(t)\,\mathrm{d}t$$

$$\psi_i^{jm}(t) = (x_i^m(t) - \bar{x}_i^m)(y_m^j(t) - \bar{y}_m^j)$$

The empirical cross Gramian is the only empirical Gramian that can be computed column-wise, because it is actually not a Gramian matrix!

[9] C. H. **Combined State and Parameter Reduction for Nonlinear Systems with an Application in Neuroscience**. Sierke Verlag Göttingen, 2017.

Low-Rank Empirical Cross Gramian

Aim: Truncated SVD of Empirical Cross Gramian:

$$W_X \overset{\text{SVD}}{=} UDV \to U_1$$

If only I could . . .

- . . . compute an SVD for a few columns at a time.
- . . . enforce a projection error for U_1.
- . . . guarantee correctness.

Enter HAPOD

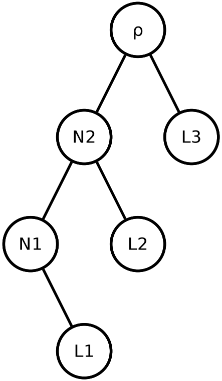
Hierarchical Approximate Proper Orthogonal Decomposition[10](HAPOD):

$$
\begin{aligned}
\hat{u}_0 &:= \{\}, \\
[\omega_s, \hat{u}_{s-1}] &\overset{\mathrm{SVD}}{=} u_s d_s v_s \to \hat{u}_s := u_s \hat{d}_s, \\
\hat{d}_{s,ii} &= \begin{cases} d_{s,ii} & d_{s,ii} < \varepsilon^2 K_s \sqrt{\frac{\sum_{j=1}^{s} K_j}{S}} \\ 0 & \text{else} \end{cases} \\
U_1 &:= \hat{u}_S \to V_1 = U_1^\intercal.
\end{aligned}
$$

- Incremental HAPOD variant
- Proper Orthogonal Decomposition (POD)
  - Basically the argument's left singular values

[10]HAPOD Matlab implementation: `http://git.io/hapod`

Memory-Bound Computation



- Compute as many columns as fit into memory as leafs L.
- Compute Sub-POD at nodes N.
- Go up the tree.
- The root POD ρ yields the HAPOD.

- Partitions distributed as leafs of some rooted tree.
- Natural per depth level parallelization.
- Projection-error-based, NOT rank-based.
- Rigorous and tight error and mode bounds.
- Only low-rank quantities are communicated.
- Compute POD, SVD or PCA.
- Use custom SVD implementation for “local” PODs.
- Relaxation parameter (accuracy vs speed)

[11] C. Himpe, T. Leibner and S. Rave. **Hierarchical Approximate Proper Orthogonal Decomposition**. arXiv e-prints math.NA: 1607.05210, 2017.

Model and Reduction

Hyperbolic Network Model (HNM)[13]:

$$\dot{x}(t) = A \tanh(Kx(t)) + Bu(t)$$
$$y(t) = Cx(t)$$

- $\dim(x(t)) = 1024$
- $\dim(u(t)) = \dim(y(t)) = 1$

Numerical Experiment:

1. Compute empirical cross Gramian columns via `emgr`
2. Compute left singular vectors via `HAPOD`

[13] Y. Quan, H. Zhang and L. Cai. **Modeling and Control Based on a New Neural Network Model**. Proceedings of the American Control Conference 3: 1928–1929, 2001.

emgr - EMpirical GRamian Framework (Version: 5.1)

Empirical Gramians:

- Empirical Controllability Gramian
- Empirical Observability Gramian
- Empirical Linear Cross Gramian
- Empirical Cross Gramian
- Empirical Sensitivity Gramian
- Empirical Identifiability Gramian
- Empirical Joint Gramian

Features:

- Interfaces for: Solver, inner product kernels & distributed memory
- Non-Symmetric option for all cross Gramians
- Compatible with OCTAVE and MATLAB
- Vectorized and parallelizable
- Open-source licensed

More info: `http://gramian.de`

C. Himpe. **emgr - The empirical Gramian Framework**. arXiv (cs.MS): 1611.00675, 2016. URL http://arxiv.org/abs/1611.00675.

Power Awareness

Single-Board Computers (SBC):

- Powerful CPU, but limited memory.
- Run Octave (on Linux).
- Low (per unit) cost.

Model Reduction:

- Online phase (Evaluation) on SBC
- **Offline** phase (Assembly) on SBC
- Less model, less power (see `PACO15`[12])

[12] C. Himpe and M. Ohlberger. **Zero-Copy Empirical Gramians**. Workshop on Power Aware Computing (PACO), 2015. `http://himpe.science/talks/himpe15_paco.pdf`

Silicon Zoo

Hardware Test Subjects:

1. **Custom PC** (`http://products.amd.com/...`):
   CPU: AMD APU (Kaveri / Steamroller)
   RAM: 32GB DDR3
2. **HP EliteDesk 705 G3 Mini** (`http://www8.hp.com/...`):
   CPU: AMD APU (Bristol Ridge / Excavator)
   RAM: 32GB DDR4
3. **NanoPi NEO** (`http://nanopi.io/nanopi-neo.html`):
   CPU: Allwinner ARM (Cortex v7)
   RAM: 0.5GB DDR3
4. **NanoPi NEO2** (`http://nanopi.io/nanopi-neo2.html`):
   CPU: Allwinner ARM (Cortex v8A)
   RAM: 0.5GB DDR3

x86 Test Systems

AMD	A10-7800	A12-9800**E**
Arch	x86-64(64-bit)	x86-64(64-bit)
Cores	4	4
Clock	3.5 Ghz	3.1 Ghz
Turbo	3.9 Ghz	3.8 Ghz
L1i-Cache	2 x 96 KB	2 x 96 KB
L1d-Cache	4 x 16 KB	**4 x 32 KB**
L2-Cache	**2 x 2 MB**	2 x 1 MB
SIMD	2x AVX+FMA3|4	2x **AVX2**+FMA3|4

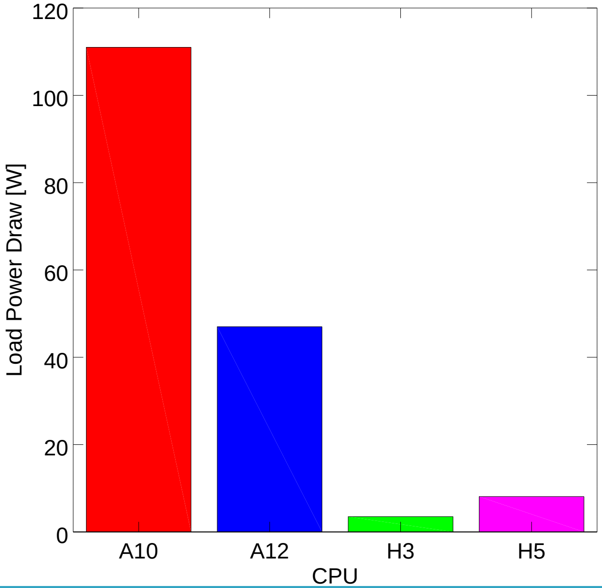
- Performance vs size optimized routing
- Further instruction set extensions: MOVBE, BMI2, ...

ARM Test Systems

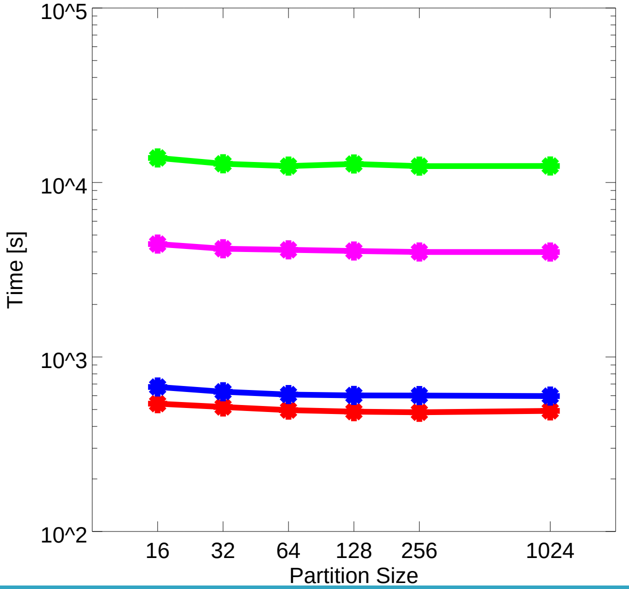
Allwinner	H3	H5
Arch	Cortex-A7(32-bit)	**Cortex-A53(64-bit)**
Cores	4	4
Clock	0.4 Ghz	0.4 Ghz
Turbo	0.8 Ghz	0.7 Ghz
L1i-Cache	4 x 32 KB	4 x 32 KB
L1d-Cache	4 x 32 KB	4 x 32 KB
L2-Cache	512 KB	512 KB
SIMD	NEON+VFP4	NEON+VFP4

- Instruction-Sets: V7 vs V8A
- Turbo set so temperature for same cooler is safe.

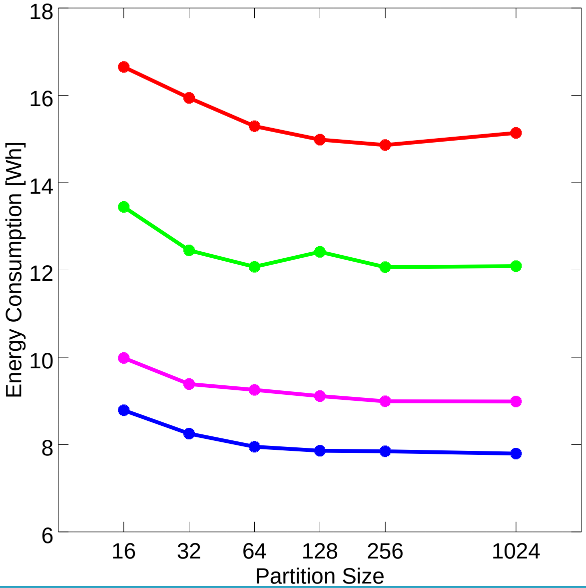
Power Draw Under Load



Experiment Runtime



Efficiency



Energy Consumption [Wh]
18
16
14
12
10
8
6
16
32
64
128
256
1024
Partition Size

Conclusion | Outlook

On the NanoPi NEO:

- Cooling is paramount!
- NEO2 can be pretty fast.
- Next: Mini cluster of 4+1 Pis.

Technicalities:

- Heuristic optimization optimization.
- `-O3` & `-ffast-math` (http://youtu.be/w5Z4JlMJ1VQ)
  - NEON SIMD requires `-ffast-math`.
- Abstract parallelization.
- A12 has an iGPU with $2:1$ single-double ratio.

Summary

- The empirical cross Gramian is special.
- HAPOD allows low-rank SVD.
- SBCs are almost there.

Extendend Abstract: doi:10.5281/zenodo.814498

Acknowledgment:
Supported by the German Federal Ministry for Economic Affairs and Energy, in the joint project: "**MathEnergy** – Mathematical Key Technologies for Evolving Energy Grids", sub-project: Model Order Reduction (Grant number: 0324019**B**).