True orbit simulation of dynamical systems and its computational complexity

Christoph Spandl

Fakultät für Informatik
Universität der Bundeswehr München

SCAN 2014

Case study: classical molecular dynamics

- Molecules (e.g. proteins) are modelled by *classical particles*. The motion of the particles is governed by *Newton's 2nd law*:

$$\vec{F} = m \cdot \vec{a}$$

- Since the acceleration $\vec{a}(t) = \dot{\vec{v}}(t)$ is the derivative of the velocity, and the velocity $\vec{v}(t) = \dot{\vec{q}}(t)$ is the derivative of the coordinate $\vec{q}(t)$,
- an MD simulation can be seen as ***solving an ODE numerically***:

$$\ddot{\vec{q}} = \frac{1}{m} \cdot \vec{F}(\vec{q}).$$

MD simulation in practice

- The ODE is ***discretized***, i.e. transformed to a *discrete dynamical system*.
- The standard integration scheme in MD is the *velocity Verlet algorithm*:

$$\begin{aligned} \boldsymbol{v}_{n+1/2} &= \boldsymbol{v}_n + \frac{h}{2m}\boldsymbol{F}(\boldsymbol{q}_n) \\ \boldsymbol{q}_{n+1} &= \boldsymbol{q}_n + h \cdot \boldsymbol{v}_{n+1/2} \\ \boldsymbol{v}_{n+1} &= \boldsymbol{v}_{n+1/2} + \frac{h}{2m}\boldsymbol{F}(\boldsymbol{q}_{n+1}). \end{aligned}$$

- ***Approximation***: integration is replaced by iteration.
- ***Crucial parameter***: step size h.

Interpretation of simulation results

The situation

- The dynamics of typical simulations show sensitive dependence on the initial condition: *Lyapunov instability*.
- The simulation scheme ***inherits*** this instability.
- Simulation times are long compared to the Lyapunov time:
- True orbits and simulated orbits may differ extremely.

Interpretation

- Not ***true orbits*** are of interest in a simulation but ***statistically defined quantities*** of the system.
- It is assumed that a simulated ***pseudo orbit*** is good enough,
- motivated by *shadowing*.

Objections

Quotation

In D. FRENKEL AND B. SMIT, *Understanding Molecular Simulation: From Algorithms to Applications*, 2nd ed., p. 73 we find:

"Hence, our trust in Molecular Dynamics Simulation ... is based largely on belief. To conclude this discussion, let us say that there is clearly still a corpse in the closet. We believe this corpse will not haunt us, and we quickly close the closet."

Exact real arithmetic

- To support this belief (or to falsify it) it is desirable to simulate true orbits. Here, this is done using the iRRAM "machine" (interactive real-RAM, [Mu00]), an implementation of the concept of the feasible real-RAM [BH98], which
  1. simulates a RAM register machine,
  2. where each register holds a real number.
  3. The simulation is done by a Turing machine
  4. and works ***iteratively*** with finite,
  5. but ***arbitrary precision***.
- The iRRAM is a software package, written in C++.
- The essential data type is `REAL`, a class with an ***arbitrary precision floating point number*** and a fixed precision floating point number representing an ***upper bound on the error*** as core elements.

Example of a discrete dynamical system

Hénon's area-preserving map

Consider the ***discrete dynamical system*** $f : M \to M$,

$$x_{n+1} = cx_n - (1-c)x_n^2 - y_n$$
$$y_{n+1} = x_n - cx_{n+1} + (1-c)x_{n+1}^2$$

with $M = \mathbb{R}^2$ and control parameter $c \in [-1, 1]$.

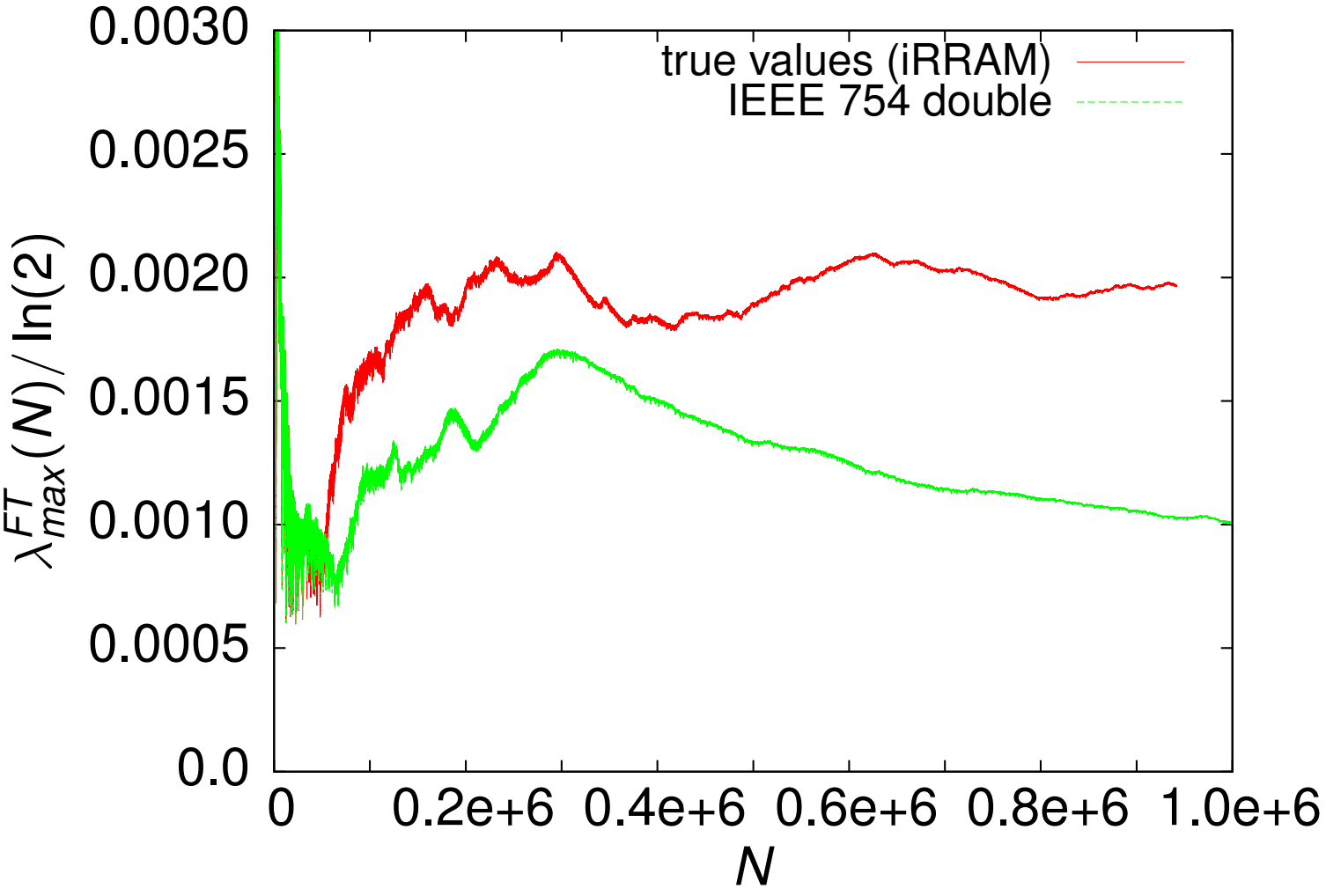
Simulation parameter

$$c = 0.24, \quad x_0 = -0.38484, \quad y_0 = 0.0$$

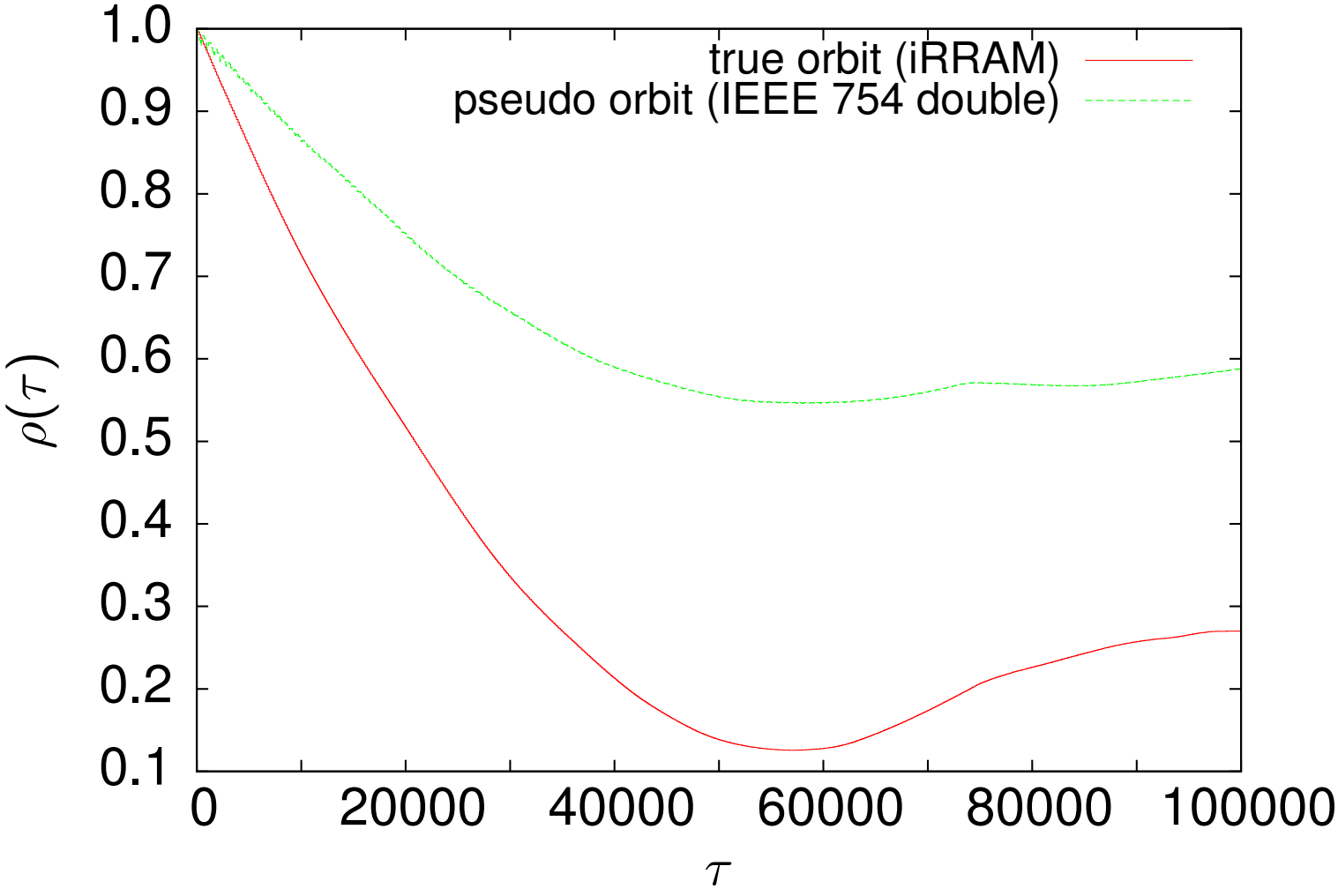
Finite time maximum Lyapunov exponent

$$\lambda_{\max}^{FT}(\boldsymbol{x}, N) = \frac{1}{N} \cdot \ln ||(Df^N)(\boldsymbol{x})||.$$

Finite time maximum Lyapunov exponent



Autocorrelation function



Complexity analysis

Loss of significance

- The loss of significant bits per iteration, $\sigma(\boldsymbol{x}, N)$, is 1.4 for the iRRAM in the above example.
- A theoretical analysis suggests an optimal $\sigma(\boldsymbol{x}, N)$ ***asymptotically*** given by the ***maximum Lyapunov exponent***:

$$\sigma(\boldsymbol{x}, N) = \frac{1}{\ln(2)} \cdot \max(0, \lambda^{FT}_{\max}(\boldsymbol{x}, N))$$

  as $N \to \infty$.
- The above calculations would expect an optimal loss of significant bits per iteration of 0.002.
- This would mean in the above example (10^6 iterations): 2000 bits of precision would suffice, instead of 1400000 bits actually used.

For interested persons

The iRRAM C++ package of Norbert Müller can be downloaded at

`https://github.com/norbert-mueller/iRRAM`

Lyapunov instability

Linear stability analysis

- Consider a small sphere S and apply f N times.
- The image of S under f^N is ***approximated*** by ***linearization***:

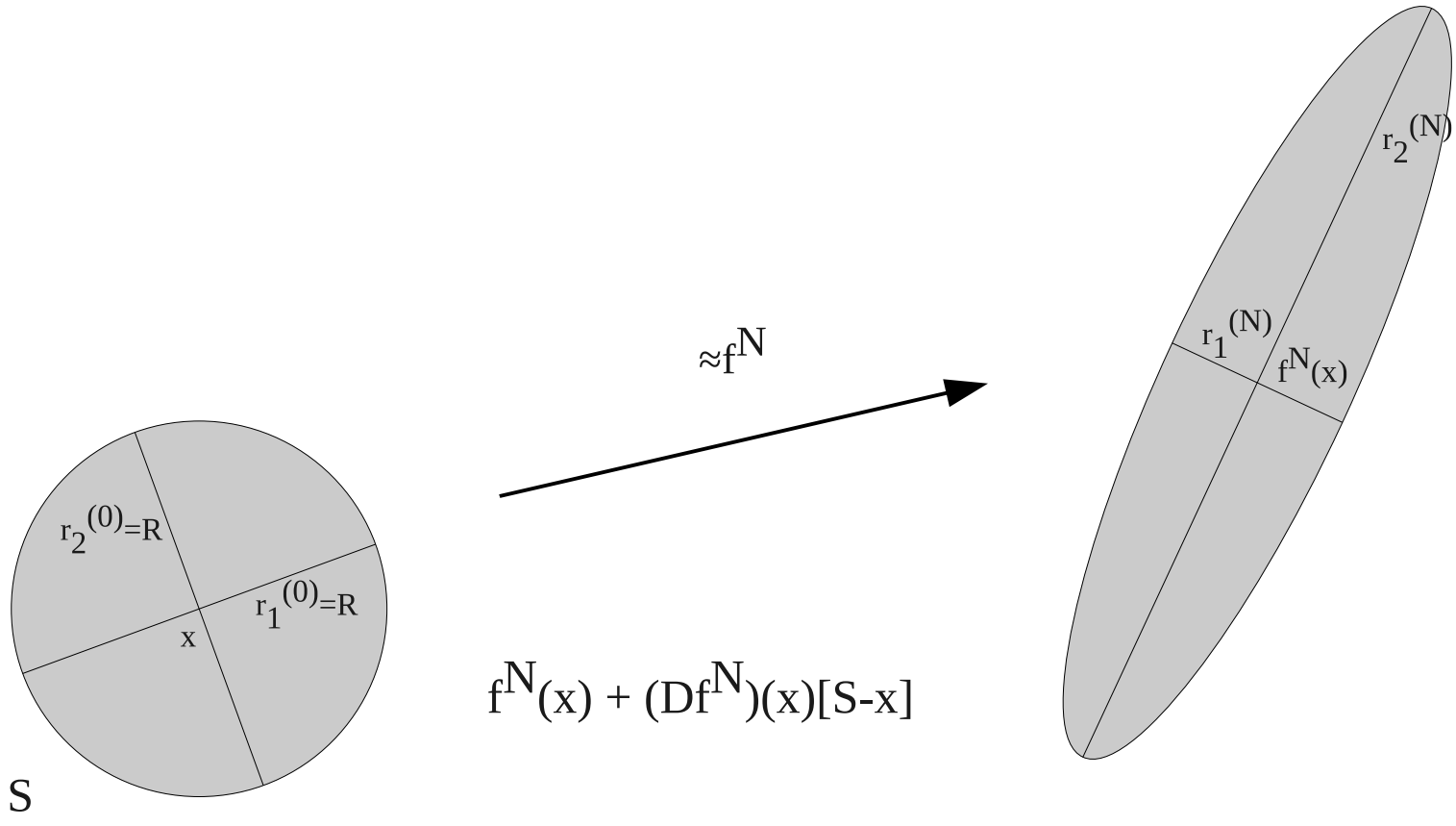
$$f^N(S) \approx f^N(x) + (Df^N)(x) \cdot (S - x).$$

- The length of the axes of the ***ellipsoid*** $(Df^N)(x) \cdot (S - x)$ are denoted by $r_1^{(N)}, \ldots, r_n^{(N)}$.
- The value $r_i^{(N)}$ measures the ***contraction*** or ***expansion*** of the dynamics.

Lyapunov exponents

The i-th *Lyapunov exponent*: $\lambda_i = \lim_{N\to\infty} \frac{1}{N} \cdot \ln(r_i^{(N)})$.

Lyapunov instability: a picture



The cocycle / triangularization

Linearized map: *cocycle*

$$x^{(k+1)} = f(x^{(k)})$$
$$z^{(k+1)} = (Df)(x^{(k)}) \cdot z^{(k)}$$
$$x^{(0)} \in M, \quad z^{(0)} = \mathbb{1}$$

QR-decomposition: Triangularization

$$Q^{(k+1)} R^{(k+1)} = (Df)(x^{(k)}) Q^{(k)}$$
$$Q^{(0)} = \mathbb{1}$$
$$\Downarrow$$
$$x^{(k+1)} = f(x^{(k)})$$
$$T^{(k+1)} = R^{(k+1)} \cdot T^{(k)}$$
$$x^{(0)} \in M, \quad T^{(0)} = \mathbb{1}$$
$$z^{(k)} = Q^{(k)} \cdot T^{(k)}$$

Connection to Lyapunov exponents: $T_{ii}^{(N)} \sim e^{\lambda_i \cdot N}$ as $N \to \infty$

Exact stability analysis (centered forms)

- Consider *boxes*

$$I = [a_1, b_1] \times \cdots \times [a_n, b_n] \subseteq \mathbb{R}^n.$$

- Each box $I \in \mathbb{IR}^n$ is uniquely represented by

$$I = x + [-1, 1] \cdot e$$

- with ***center*** $x = \mathrm{mid}(I) \in \mathbb{R}^n$ and ***extent*** $e = \frac{1}{2}\mathrm{wid}(I) \in \mathbb{R}^n_+$.
- Then (assume $f \in C^2$) the inclusion follows (*centered form*):

$$f(I) \subseteq f(x) + W(I, x) \cdot (I - x)$$
$$W(I, x) = (Df)(x) + \tfrac{1}{2}(D^2 f)(I) \cdot (I - x).$$

- Finally use Lipschitz bound $|(D^2 f)(I) \cdot (I - x)| \leq \|I - x\|_\infty L \in \mathbb{R}^{n \times n}$.

The inclusion cocycle

Dynamics of boxes: *inclusion cocycle*

$$
\begin{aligned}
x^{(k+1)} &= f(x^{(k)}) \\
z^{(k+1)} &= |V(x^{(k)}, e^{(k)})| \cdot z^{(k)} \\
x^{(0)} &\in M, \quad z^{(0)} = \mathbb{1} \\
e^{(k)} &= z^{(k)} \cdot e^{(0)}, \quad e^{(0)} \in \mathbb{R}^n_+
\end{aligned}
$$

Triangularization

$$
\begin{aligned}
P^{(k+1)} S^{(k+1)} &= V(x^{(k)}, e^{(k)}) P^{(k)} \\
P^{(0)} &= \mathbb{1} \\
&\Downarrow \\
x^{(k+1)} &= f(x^{(k)}) \\
T^{(k+1)} &= |S^{(k+1)}| \cdot T^{(k)} \\
x^{(0)} &\in M, \quad T^{(0)} = \mathbb{1} \\
e^{(k)} &= |P^{(k)}| \cdot T^{(k)} \cdot e^{(0)}
\end{aligned}
$$

where $V(x, e) = (Df)(x) + \frac{1}{2}\|e\|_\infty \operatorname{sgn}((Df)(x)) : L$.

From the inclusion cocycle to an algorithm

The computational model

Real numbers $x \in \mathbb{R}$ are represented by a

1. fixed point approximation $\hat{x} \in \widehat{\mathbb{R}}$ and an
2. upper bound $\overline{e} \in \widehat{\mathbb{R}}_+$ on the error.

The iteration algorithm

$$
\begin{aligned}
\hat{x}^{(k+1)} &= \hat{f}(\hat{x}^{(k)}) \\
\hat{T}^{(k+1)} &= (|\hat{S}^{(k+1)}| + c \cdot U) \cdot \hat{T}^{(k)} + \mathbb{1} \\
\hat{x}^{(0)} &\in M \cap \widehat{\mathbb{R}}^n(p^s), \quad \hat{T}^{(0)} = \mathbb{1} \\
\overline{e}^{(k)} &= (|\hat{P}^{(k)}| + d \cdot E) \cdot \hat{T}^{(k)} \cdot \beta^{-p^s}.
\end{aligned}
$$