

Recent Advances in the Rigorous Integration of Flows of ODEs with Taylor Models

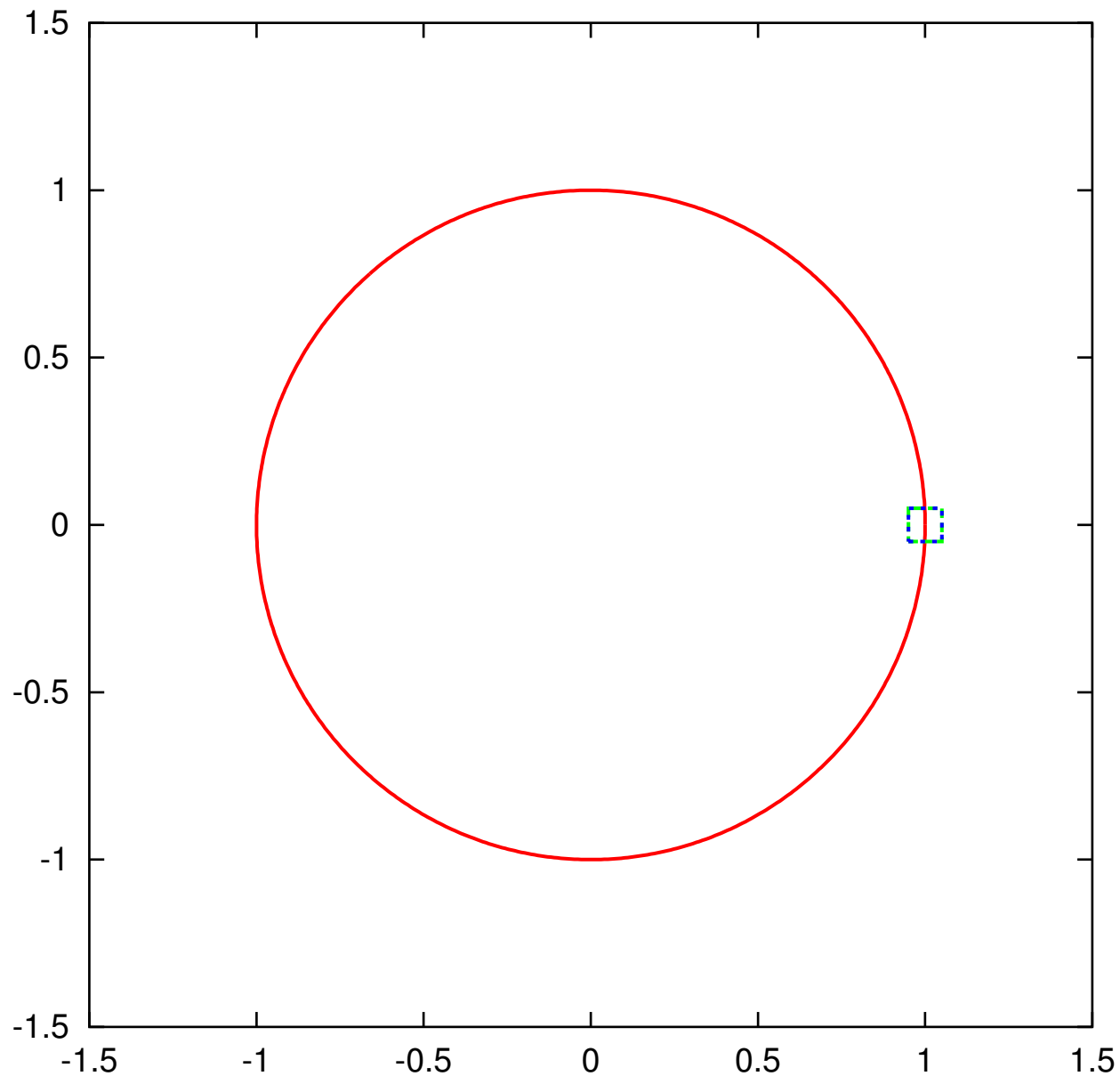
Kyoko Makino and Martin Berz

Department of Physics and Astronomy
Michigan State University

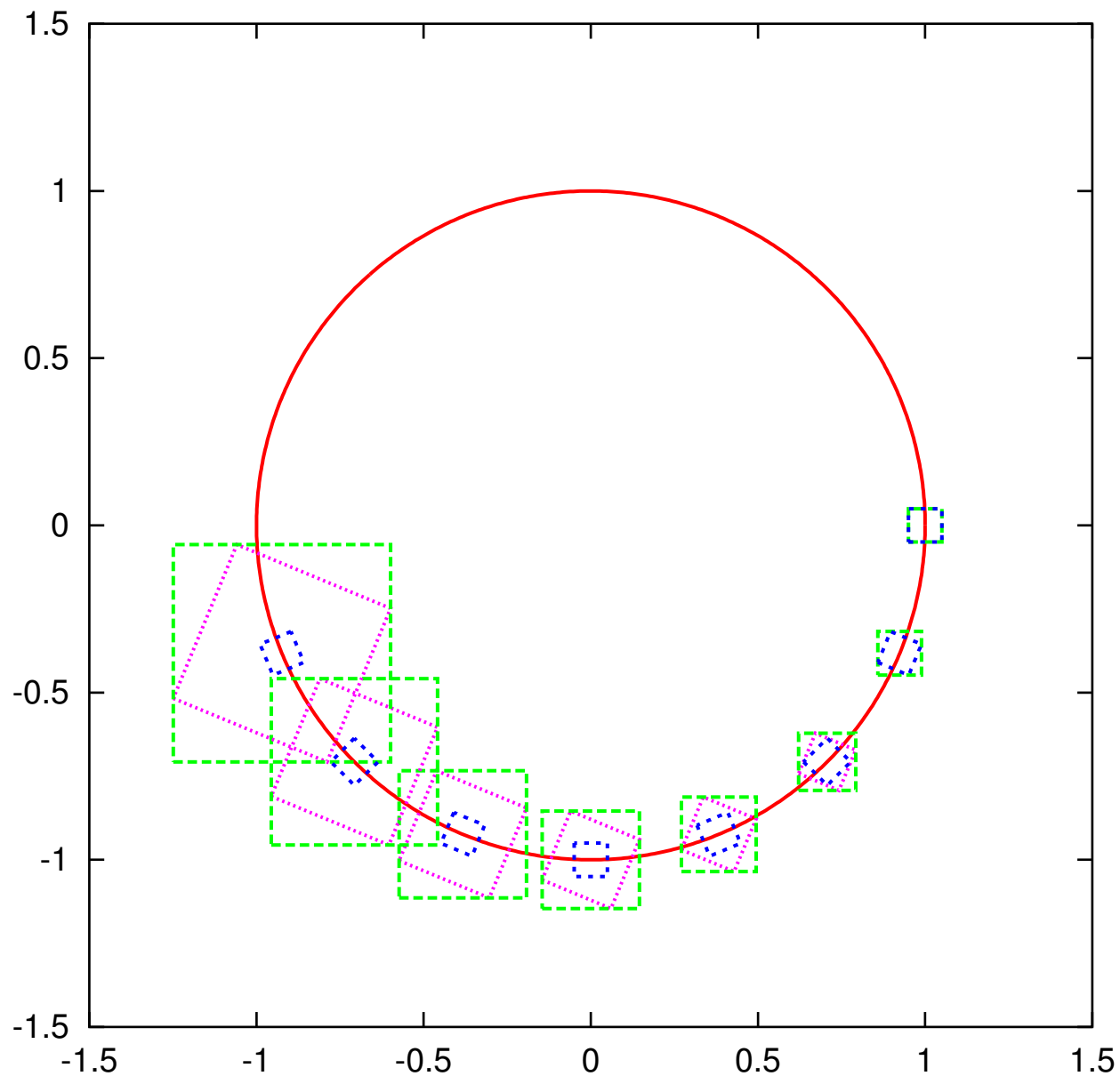
Outline

1. Review of the old version of COSY-VI
2. The Reference Trajectory and the Flow Operator
3. Step Size Control
4. Error Parametrization of Taylor Models
5. Dynamic Domain Decomposition
6. Examples

To transport a large phase space volume with validation,



Over Estimation has to be controlled.



Key Features and Algorithms of COSY-VI

- High order expansion not only in time t but also in transversal variables \vec{x} .
- Capability of weighted order computation, allowing to suppress the expansion order in transversal variables \vec{x} .
- Shrink wrapping algorithm including blunting to control ill-conditioned cases.
- Pre-conditioning algorithms based on the Curvilinear, QR decomposition, and blunting pre-conditioners.
- Resulting data is available in various levels including graphics output.

The Volterra Equation

Describe dynamics of two conflicting populations

$$\frac{dx_1}{dt} = 2x_1(1 - x_2), \quad \frac{dx_2}{dt} = -x_2(1 - x_1)$$

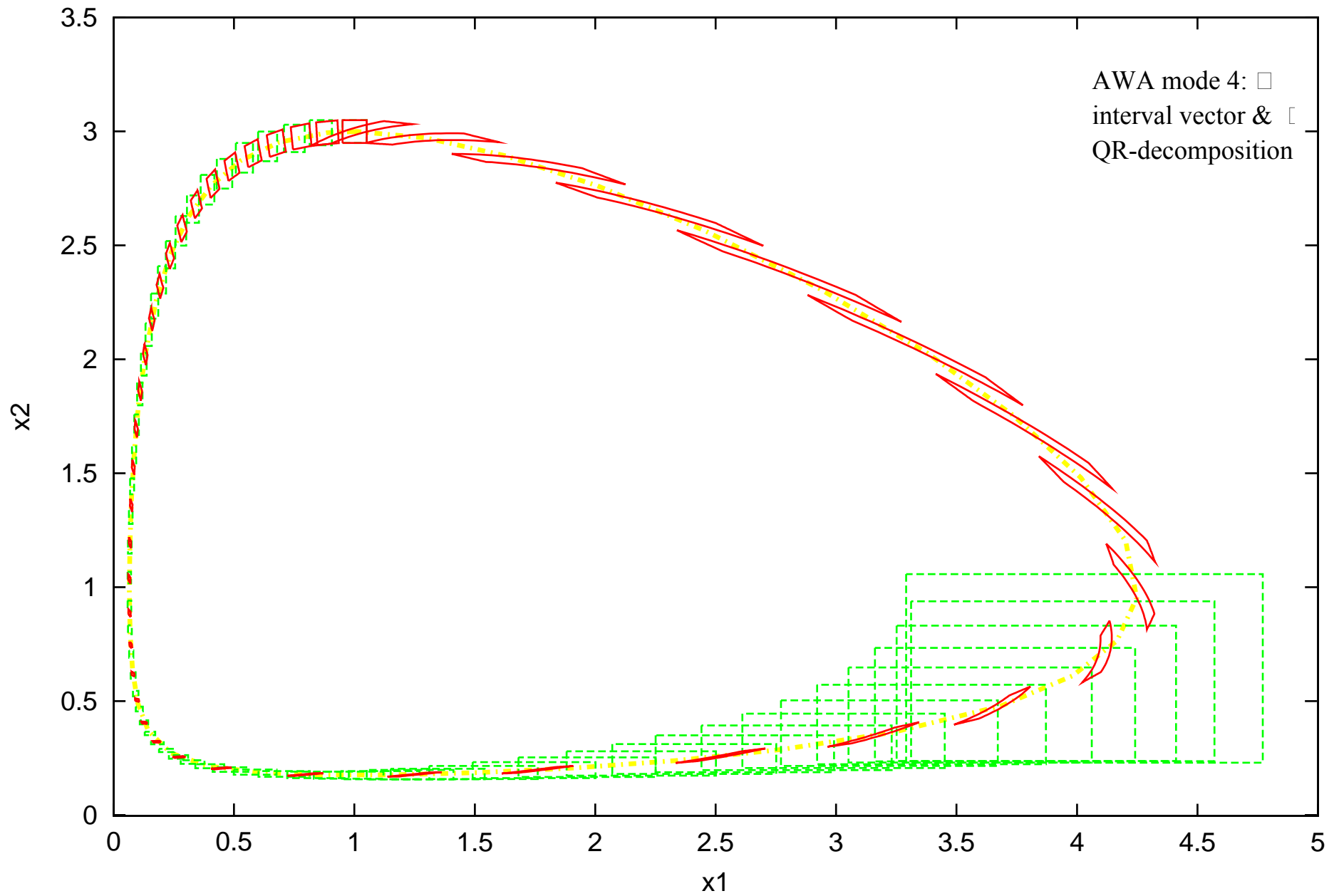
Interested in initial condition

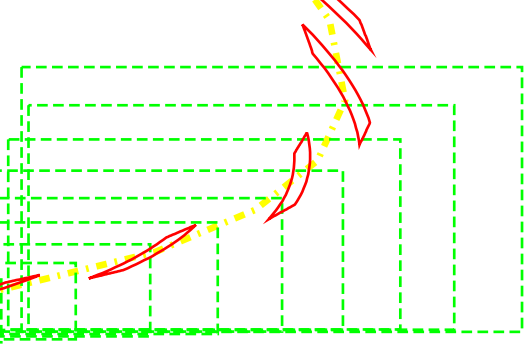
$$x_{01} \in 1 + [-0.05, 0.05], \quad x_{02} \in 3 + [-0.05, 0.05] \quad \text{at } t = 0.$$

Satisfies constraint condition

$$C(x_1, x_2) = x_1 x_2^2 e^{-x_1 - 2x_2} = \text{Constant}$$

Integration of the Volterra eq. COSY-VI and AWA





Error Parametrization of Taylor models

Motivation: Is it possible to absorb the remainder error bound intervals of Taylor models into the polynomial parts using additional parameters?

Phrase the question as the following problem:

1. Have Taylor models with 0 remainder error interval, which depend on the independent variables \vec{x} and the parameters $\vec{\alpha}$.

$$\vec{T}_0 = \vec{P}_0(\vec{x}, \vec{\alpha}) + \overrightarrow{[0, 0]}.$$

2. Perform Taylor model arithmetic on \vec{T}_0 , namely $\vec{F}(\vec{T}_0)$

$$\vec{F}(\vec{T}_0) = \vec{P}(\vec{x}, \vec{\alpha}) + \vec{I}_F, \text{ where } \vec{I}_F \neq \overrightarrow{[0, 0]}.$$

3. Try to absorb \vec{I}_F into the polynomial part that depends on $\vec{\alpha}$

$$\vec{P}(\vec{x}, \vec{\alpha}) + \vec{I}_F \subseteq \vec{P}'(\vec{x}, \vec{\alpha}) + \overrightarrow{[0, 0]}. \quad (\text{A})$$

Observe

$$\vec{P}(\vec{x}, \vec{\alpha}) = \underbrace{\vec{P}(\vec{x}, 0)}_{\vec{\alpha}\text{-indep.}} + \underbrace{\vec{P}(\vec{x}, \vec{\alpha}) - \vec{P}(\vec{x}, 0)}_{\vec{\alpha}\text{-dependent}} = \vec{P}(\vec{x}, 0) + \vec{P}_\alpha(\vec{x}, \vec{\alpha})$$

The size of $\vec{P}(\vec{x}, 0)$ is much larger than the rest, because the rest is essentially errors. The process of (A) does not alter $\vec{P}(\vec{x}, 0)$, so set the $\vec{\alpha}$ -independent part $\vec{P}(\vec{x}, 0)$ aside from the whole process, which helps the numerical stability of the process.

The task is now

$$\vec{P}_\alpha(\vec{x}, \vec{\alpha}) + \vec{I}_F \subseteq \vec{P}'_\alpha(\vec{x}, \vec{\alpha}) + \overrightarrow{[0, 0]}.$$

We limit $\vec{P}_\alpha(\vec{x}, \vec{\alpha})$ to be only **linearly** dependent on $\vec{\alpha}$.

$$\vec{P}_\alpha(\vec{x}, \vec{\alpha}) + \vec{I}_F = \left(\widehat{M} + \widehat{M}(\vec{x}) \right) \cdot \vec{\alpha} + \vec{I}_F.$$

Express \vec{I}_F by the matrix form using additional parameters $\vec{\beta}$

$$\vec{I}_F \subseteq \left(\widehat{I}_F + \widehat{I}_F(\vec{x}) \right) \cdot \vec{\beta}.$$

where $\widehat{I}_F(\vec{x}) = 0$ and $\left(\widehat{I}_F \right)_{ii} = |I_{Fi}|$.

$$\vec{P}_\alpha(\vec{x}, \vec{\alpha}) + \vec{I}_F \subseteq \left(\widehat{M} + \widehat{M}(\vec{x}) \right) \cdot \vec{\alpha} + \left(\widehat{I}_F + \widehat{I}_F(\vec{x}) \right) \cdot \vec{\beta}.$$

View this as a collection of $2 \cdot v$ column vectors associated to $2 \cdot v$ parameters $\vec{\alpha}$ and $\vec{\beta}$. Recall a matrix, or a collection of v column vectors, represent a parallelepiped. The problem is now to find a **set sum of two parallelepipeds**.

Psum Algorithm for choosing column vectors

Task: Choose v vectors out of n vectors \vec{s}_i , $i = 1, \dots, n$, $n \geq v$.

1. Choose the longest vector \vec{s}_k , and assign it as \vec{t}_1 . Normalize it as $\vec{e}_1 = \vec{t}_1 / |\vec{t}_1|$.
2. Out of the remaining vectors \vec{s}_i , choose the j -th vector $\vec{t}_j = \vec{s}_k$ such that

$$\frac{|\vec{s}_k|^2 - \sum_{m=1}^{j-1} |\vec{s}_k \cdot \vec{e}_m|^2}{|\vec{s}_k|^{2p}}$$

is largest. Compute \vec{e}_j , the orthonormalized vector of \vec{t}_j to $\vec{e}_1, \dots, \vec{e}_{j-1}$. (Gram-Schmidt)

3. Repeat the process 2 until $j = v$.

Experimentally, $p = 0.5$ is found to be efficient and robust for obtaining a set sum of two parallelepipeds

Psum Algorithm for two parallelepipeds

Task: Obtain a set sum of two parallelepipeds \widehat{M}_1 and \widehat{M}_2 .

1. Prepare the basis \widehat{M}_b using the Psum algorithm for choosing v column vectors out of $2 \cdot v$ column vectors from \widehat{M}_1 and \widehat{M}_2 .
2. Compute conditioned parallelepipeds $\widehat{M}_b^{-1} \cdot \widehat{M}_1$ and $\widehat{M}_b^{-1} \cdot \widehat{M}_2$.
3. Confine the conditioned parallelepipeds by bounding them.

$$\vec{B}_1 = \text{bound} \left(\widehat{M}_b^{-1} \cdot \widehat{M}_1 \right) \text{ and } \vec{B}_2 = \text{bound} \left(\widehat{M}_b^{-1} \cdot \widehat{M}_2 \right).$$

4. Compute the interval sum $\vec{B} = \vec{B}_1 + \vec{B}_2$. \vec{B} confines the conditioned set sum of the conditioned parallelepipeds.

5. From \vec{B} , set up a parallelepiped as a box $\widehat{B} = \begin{pmatrix} |B_1| & & 0 \\ & \dots & \\ 0 & & |B_v| \end{pmatrix}$.

6. Compute $\widehat{M}_b \cdot \widehat{B}$, which is a set sum of \widehat{M}_1 and \widehat{M}_2 under \widehat{M}_b .

Error Absorption

We now chose a favoured collection of v column vectors $\widehat{L} + \widehat{\widehat{L}}(\vec{x})$ using the Psum algorithm. Collect the left over v column vectors to $\widehat{E} + \widehat{\widehat{E}}(\vec{x})$. Associate them to $2 \cdot v$ parameters $\vec{\alpha}'$ and $\vec{\beta}'$.

$$\vec{P}_\alpha(\vec{x}, \vec{\alpha}) + \vec{I}_F \subseteq \left(\widehat{L} + \widehat{\widehat{L}}(\vec{x}) \right) \cdot \vec{\alpha}' + \left(\widehat{E} + \widehat{\widehat{E}}(\vec{x}) \right) \cdot \vec{\beta}'.$$

Since $\vec{\alpha}'$ and $\vec{\beta}'$ do not appear anymore, we can rename $\vec{\alpha}'$ and $\vec{\beta}'$ as $\vec{\alpha}$ and $\vec{\beta}$ for the simplicity.

$$\begin{aligned} \vec{P}_\alpha(\vec{x}, \vec{\alpha}) + \vec{I}_F &\subseteq \left(\widehat{L} + \widehat{\widehat{L}}(\vec{x}) \right) \cdot \vec{\alpha} + \left(\widehat{E} + \widehat{\widehat{E}}(\vec{x}) \right) \cdot \vec{\beta} \\ &= \widehat{L} \circ \left[\widehat{L}^{-1} \circ \left(\widehat{L} + \widehat{\widehat{L}}(\vec{x}) \right) \cdot \vec{\alpha} + \widehat{L}^{-1} \circ \left(\widehat{E} + \widehat{\widehat{E}}(\vec{x}) \right) \cdot \vec{\beta} \right] \\ &\subseteq \widehat{L} \circ \left[\left(\widehat{I} + \widehat{L}^{-1} \circ \widehat{\widehat{L}}(\vec{x}) \right) \cdot \vec{\alpha} + \widehat{B} \cdot \vec{\beta} \right] \end{aligned}$$

where \widehat{B} is a diagonal matrix with the i -th element is $|B_i|$ and $\vec{B} = \text{bound} \left(\widehat{L}^{-1} \circ \left(\widehat{E} + \widehat{\widehat{E}}(\vec{x}) \right) \cdot \vec{\beta} \right)$.

If the **diagonal terms** of $\left(\widehat{I} + \widehat{L}^{-1} \circ \widehat{L}(\vec{x})\right)$ are **positive**,

$$\begin{aligned} \vec{P}_\alpha(\vec{x}, \vec{\alpha}) + \vec{I}_F &\subseteq \widehat{L} \circ \left[\left(\widehat{I} + \widehat{L}^{-1} \circ \widehat{L}(\vec{x})\right) \cdot \vec{\alpha} + \widehat{B} \cdot \vec{\alpha} \right] \\ &= \widehat{L} \circ \left(\widehat{I} + \widehat{L}^{-1} \circ \widehat{L}(\vec{x})\right) \cdot \vec{\alpha} + \widehat{L} \circ \widehat{B} \cdot \vec{\alpha} \\ &= \left(\widehat{L} + \widehat{L}(\vec{x}) + \widehat{L} \circ \widehat{B}\right) \cdot \vec{\alpha}. \end{aligned}$$

Note: A modification to use \widehat{A} instead of \widehat{L} , when $\widehat{A} \approx \widehat{L}$, is done easily. This involves bounding of $\widehat{A}^{-1} \circ \left(\widehat{L} - \widehat{A}\right) \cdot \vec{\alpha}$ and the diagonal terms to be checked positive are those of $\left(\widehat{I} + \widehat{A}^{-1} \circ \widehat{L}(\vec{x})\right)$.

Cost of Additional Parameters

For a v dimensional system, we need v parameters $\vec{\alpha}$ to absorb Taylor model remainder error bound intervals. The dependence on $\vec{\alpha}$ is limited to **linear**. So, we use weighted DA. Choose an appropriate weight order w for $\vec{\alpha}$.

- The dependence on $\vec{\alpha}$ has to be kept linear. Namely $2 \cdot w > n$, where n is the computational order of Taylor models. Choose

$$w = \text{Int} \left(\frac{n}{2} \right) + 1.$$

Maximum size necessary for DA and TM for $v = 2$.

n	v	DA	TM		v	DA	TM		w	v_w	DA	TM
13	2	105	140		2 + 2	2380	2419		7	$2 + 2_w$	161	200
21	2	253	304		2 + 2	12650	12705	\Rightarrow	11	$2 + 2_w$	385	440
33	2	595	670		2 + 2	66045	66124		17	$2 + 2_w$	901	980

Dynamic Domain Decomposition

For extended domains, this is **natural equivalent** to step size control. Similarity to what's done in global optimization.

1. Evaluate ODE for $\Delta t = 0$ for current flow.
2. If resulting remainder bound R greater than ε , split the domain along variable leading to longest axis.
3. Absorb R in the TM polynomial part using the error parametrization method. If it fails, split the domain along variable leading to largest x dependence of the error.
4. Put one half of the box on stack for future work.

Things to consider:

- Utilize "First-in-last-out" stack; minimizes stack length. Special adjustments for stack management in a parallel environment, including load balancing.
- Outlook: also dynamic order control for dependence on initial conditions

The Henon Map

$$H(x, y) = (1 - ax^2 + y, bx).$$

We set the parameters $a = 1.4$ and $b = 0.3$, which are originally considered by Henon. The map H has two fixed points.

$$\vec{p}_1 = (0.63135, 0.18940) \quad \text{and} \quad \vec{p}_2 = (-1.13135, -0.33941).$$

henon: Number of Objects

To carry out multiple mappings of the Henon map, Taylor model objects underwent the domain decomposition.

Number of Taylor model objects used for multiple mappings:

	n	w	for 5 steps	for 7 steps
box1	33	17	3	1386
box2	21	11	148	1691
box3	33	17	8	2839