Multiple precision computation of exponentially small splittings (Lecture 2)

Rafael Ramírez-Ros

(Available at `http://www.ma1.upc.edu/~rafael/research.html`)

`Rafael.Ramirez@upc.edu`

Universitat Politècnica de Catalunya

Mission statements

- Present the exponentially small splitting problem for analytic area-preserving maps. [Lecture 1]
- Explain the computational challenges of this problem. [Lecture 1]
- Give some general principles to improve the efficiency of any computation that requires the use of a multiple precision arithmetic. [Lecture ?]
- Learn how to compute the Lazutkin homoclinic invariant in the general case. [Lecture 2]
- Implement explicitely the simplest case: the Hénon map. [Lecture 2]

Notations (1/2)

- M is the bi-dimensional phase space
- Ω is the area form.
- $f : M \to M$ is the analytic weakly-hyperbolic area-preserving map.
- $R : M \to M$ is the reversor.
- $\operatorname{Fix} R = \{m \in M : G(m) = 0\}$ is the symmetry line of the reversor.
- $m_\infty \in M$ is the saddle point.
- $\lambda \gtrsim 1$ is the characteristic multiplier.
- $h = \log \lambda \ll 1$ is the characteristic exponent.
- $W^{\pm}$ are the stable and unstable invariant curves of the saddle point.

Notations (2/2)

- $m : \mathbb{R} \to W^+$ is the natural parameterization of the unstable curve.
- $m_0 = m(r_0)$, $r_0 > 0$, is the primary symmetric homoclinic point on $\operatorname{Fix} R$.
- $\omega = (r_0)^2 \Omega\big(\mathrm{d}R(m_0) m'(r_0), m'(r_0)\big)$ is the Lazutkin homoclinic invariant.
- $c > 0$ is the constant such that $\omega = \mathcal{O}(\mathrm{e}^{-c/h})$ as $h \to 0^+$.
- $D = m([r_1/\lambda, r_1))$, $0 < r_1 < r_0$, is a fundamental domain of W^+.
- $N \approx h^{-1} \log(r_0/r_1) = \mathcal{O}(P/Kh)$ is the smallest integer such that

$$f^N(D) \cap \operatorname{Fix} R \neq \emptyset.$$

- $\bar{r}_0 \in [r_1/\lambda, r_1)$ is the root of the one-dimensional equation

$$Z(r) := G(f^N(m(r))) = 0.$$

First big trick: Don’t fix the order

- In order to control the number of iterations $N = \mathcal{O}(P/Kh)$, the order K must increase when $h \to 0^+$.
- Orders below hundreds do not serve in edge scenarios. For sample, we shall see that the optimal choice in the Hénon map with $h = 0.02$ is $K \approx 100$.
- Therefore, we must find a recursive algorithm to determine the Taylor coefficients up to any given (but arbitrary!) order.
- It is easier to find a good algorithm for maps that have explicit expressions: the Hénon map, the Standard map, polynomial standard maps, perturbed McMillan maps, etc.
- Implicit maps can also be dealt with, although they require more work. For instance, there is a nice algorithm for the billiard maps introduced in the first Lecture.

A sample: the Hénon map

- Let $x(r) = \sum_{k\geq 1} x_k r^k$ and $y(r) = \sum_{k\geq 1} y_k r^k$ be the Taylor expansions of the natural parameterization $m(r) = (x(r), y(r))$ of the Hénon map

$$x_1 = x + y_1, \qquad y_1 = y + \epsilon x(1-x).$$

- The relation $f(m(r)) = m(\lambda r)$ is equivalent to the functional equations

$$x(\lambda r) - x(r) = y(\lambda r), \qquad y(\lambda r) - y(r) = \epsilon x(r)\big(1 - x(r)\big).$$

- We get from relation $x(\lambda r) - (2+\epsilon)x(r) + x(r/\lambda) = -\epsilon x(r)^2$ that

$$d_k x_k = -\epsilon \sum_{j=1}^{k-1} x_j x_{k-j}, \qquad \forall k \geq 1$$

  where $d_k = \lambda^k - (2+\epsilon) + \lambda^{-k}$ and $d_k = 0 \Leftrightarrow k = \pm 1$.
- Hence, x_1 is free and we normalize it by taking $x_1 = 1$.
- Next, we can compute recursively x_k for all $k \geq 2$.
- Finally, $y(\lambda r) = x(\lambda r) - x(r) \Longrightarrow y_k = (1 - \lambda^{-k}) x_k$ for any $k \geq 1$.

A couple of little tricks

- Evaluate the Taylor expansions using the *Horner's rule*.
- The computational effort to perform the *convolution*

$$\sum_{j=a}^{b-a} x_j x_{b-j} = x_a x_{b-a} + x_{a+1} x_{b-a-1} + \cdots + x_{b-a-1} x_{a+1} + x_{b-a} x_a$$

can be reduced by half using the formulae

$$\sum_{j=a}^{b-a} x_j x_{b-j} = \begin{cases} 2\sum_{j=a}^{(b-1)/2} x_j x_{b-j} & \text{if } b \text{ is odd} \\ 2\sum_{j=a}^{b/2-1} x_j x_{b-j} + \left(x_{b/2}\right)^2 & \text{if } b \text{ is even} \end{cases} .$$

Second big trick: Don’t fix the precision

- In order to find, with a high precision P, the root of a function $Z : (a, b) \rightarrow \mathbb{R}$ such that $Z(a)$ and $Z(b)$ have opposite signs, we shall apply the following algorithm:
  1. Refine the interval (a, b) with some secure method (bisection, Brent’s) in “single” precision.
  2. Choose some fast iterative method (Newton’s, Brent’s, Ridders’) and *increase the precision by a factor equal to its order of convergence after each iteration*. For instance, doubling the precision in Newton’s method.
  3. Stop the iterations when we exceed the given precision P.
  4. Don’t check the error.
- This method rocks! Really.

A silly trick: Choose the optimal "single" precision

- This previous algorithm can give the root at the cost of just

$$1 + \frac{1}{4} + \frac{1}{16} + \frac{1}{64} + \cdots = \sum_{n\geq 0} 4^{-n} = 4/3$$

  evaluations of the function $Z(r) := G(f^N(m(r)))$ with precision P.

- The idea is silly, but effective: to determine the optimal "single" precision p from a certain limited range that gives the "final" precision P with the minimum computational effort.

- Example with Newton's method: To reach $P = 4000$ from a "single" precision $p \leq 18$, we see that
  - $p = 18, 36, 72, 144, 288, 576, 1152, 2304, 4608, 9216, \ldots$
  - $p = 17, 34, 68, 136, 272, 544, 1088, 2176, 4352, 8704, \ldots$
  - $p = 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, \ldots$
  - $p = 15, 30, 60, 120, 240, 480, 960, 1920, 3840, 7680, \ldots$
  - Et cetera.

  Thus, $p = 16$ is the optimal "single" precision and $p = 15$ is the worst one.

Where are we now?

- The main numerical difficulties that appear during the study of the singular splitting of our maps are the computation of:
  - The map f and its differential with an arbitrary precision P;
  - The Taylor expansion of $m(r)$ up to an arbitrary order K; and
  - The Lazutkin homoclinic invariant ω with an arbitrary precision Q.
- Clearly, the precision Q is an input of the algorithm.
- On the contrary, P and K must be determined in an automatic way when the computation begins.

The choice of P

- We assume that $\omega = \mathcal{O}(\mathrm{e}^{-c/h})$ for some constant $c > 0$. For instance, we recall that

map	Hénon	Standard	polynomial	“McMillan”	“Billiard”
c	$2\pi^2$	π^2	variable	π^2	π^2

- Let $S \approx \frac{c}{h\log(10)}$ be the number of digits lost by cancellation.
- For the sake of safety, set $P = 1.1(Q + S)$.

The choice of r_1

- Let $\bar{m}_K(r) = \sum_{k=0}^{K} m_k r^k$ be the Taylor polynomial of degree K of the natural parameterization $m(r)$ of the unstable curve.
- Once fixed an order $K \geq 1$ and a precision P, we need a parameter $r_1 > 0$, as biggest as possible, such that

$$|m(r) - \bar{m}_K(r)| \leq 10^{-P}, \qquad \forall r \in (0, r_1).$$

- If the sequence $(m_k)_{k\geq 0}$ is alternate and $|m_k| \leq C\rho^k$ for some constants $C, \rho > 0$, then it suffices to set r_1 by means of the relation

$$C(\rho r_1)^{K+1} = 10^{-P}.$$

- These hypotheses hold for the Hénon map with $C = 1$ and $\rho = 1/5$, so we can set $r_1 = 5 \times 10^{-P/(K+1)}$.
- If the map is entire (as the Hénon map), the coefficients m_k decrease asymptotically at a factorial speed. Nevertheless, this factorial behaviour appears only at very high orders and so, it is not so useful.

The choice of K

- The order K is chosen to minimize the computation time.
- In order to determine it, we must construct a function $T = T(k)$ that is proportional to the CPU time, where the variable k runs over the range of possible orders.
- The function $T(k)$ is approximated by a sum of three terms: time to compute the Taylor expansions, time to solve the equation $Z(r) = 0$, and time to compute ω.
- For instance, using Newton's method in the Hénon map, we have that

$$T(k) \approx k^2/4 + 4N + 3N \approx k^2/4 + 7P\log(10)/kh$$

because $N \approx h^{-1}\log(r_0/r_1) = h^{-1}(\log r_0 - \log 5 + P\log(10)/(k+1)) \approx h^{-1}P\log(10)/(k+1)$, and so the optimal order is

$$K \approx \sqrt[3]{14P\log(10)/h}.$$

On the CPU time for the Hénon map

- How many “products” takes the computation of the Taylor expansion up to order K in the previous Hénon example?
  Answer: $K^2/4 + \mathcal{O}(K)$, if we use the convolution trick.
- How many “products” takes Newton’s method in the Hénon map?
  Answer: One evaluation of $\mathrm{d}f$ requires 3 products, so $4N = \frac{4}{3}3N$ (approximately).
- Once computed the root $\bar{r}_0 \in [r_1\lambda, r_1)$ that gives the homoclinic point: How many “products” takes the computation of ω in the Hénon map?
  Answer: One evaluation of $\mathrm{d}f$ requires 3 products, so $3N$ (approximately).
- *Problem:* Check that, using all the previous (big, little and silly) tricks and assuming that products in our multiple precision arithmetic take a time quadratic in P, the order of the CPU time in the Hénon problem for fixed Q can be reduced to $\mathcal{O}\left(h^{-10/3}\right)$ from the original $\mathcal{O}\left(h^{-4}|\log h|\right)$.
- *Hard Problem:* Improve this algorithm, while keeping the same multiple precision arithmetic.

The general algorithm

Given the characteristic exponent h and the desired precision Q, follow the steps:

1. Compute the number of digits $S \approx \frac{c}{h \log(10)}$ lost by cancellation.
2. Set the precision $P = 1.1(Q+S)$, by safety.
3. Choose the order K by minimizing the function $T(k)$.
4. Compute the Taylor expansion $\bar{m}(r) = \sum_{k=0}^{K} m_k r^k$.
5. Choose the biggest $r_1 > 0$ such that $|m(r) - \bar{m}(r)| \leq 10^{-P}$ for all $r \in (0, r_1)$.
6. Find the smallest integer N such that $f^N(\bar{m}([r_1/\lambda, r_1)) \cap \operatorname{Fix} R \neq \emptyset$.
7. Find the root $\bar{r}_0$ of the equation $G(f^N(\bar{m}(r))) = 0$ in the interval $[r_1/\lambda, r_1)$.
8. Compute the Lazutkin homoclinic invariant

$$\begin{aligned} \omega &= (r_0)^2 \Omega\big(\mathrm{d}R(m_0) m'(r_0), m'(r_0)\big) \\ &\approx (\bar{r}_0)^2 \Omega\big(\mathrm{d}R(f^N(\bar{m}(\bar{r}_0)))\mathrm{d}f^N(\bar{m}(\bar{r}_0))\bar{m}'(\bar{r}_0), \mathrm{d}f^N(\bar{m}(\bar{r}_0))\bar{m}'(\bar{r}_0)\big). \end{aligned}$$

9. Enjoy! (optional).

Exercises (1/2)

Write the recursions to compute the Taylor expansions of the natural paremeterizations in the following maps (in increasing order of difficulty):

- *(DS & RRR, 1999)* The perturbed McMillan map

  $$f(x,y) = \left(y, -x + 2\mu_0 y/(1+y^2) + \epsilon y^{2n+1}\right)$$

  for several "small" values of $n \geq 1$.
- *(VG & CS, 2007)* The polynomial maps $(x,y) \mapsto (x + y + \epsilon p(x), y + \epsilon p(x))$ for several "simple" polynomials or rational functions $p(x)$.
- *(CS, 20??)* The Standard map $(x,y) \mapsto (x + y + \epsilon \sin x, y + \epsilon \sin x)$.
- *(RRR, 2005)* The billiard maps associated to the perturbed ellipses

  $$C = \left\{ (x,y) \in \mathbb{R}^2 : x^2 + \frac{y^2}{1-e^2} + \epsilon (ey)^{2n} = 1 \right\}$$

  for several "small" values of $n \geq 2$.

Exercises (2/2)

- Estimate the order of the general algorithm for all of the previous maps.
- Implement this algorithm in some platform (GMP, PARI/GP, real men) for some of the previous maps.
- Write a paper describing and improving the general algorithm and estimate explicitely its cost in terms of the cost of one evaluation of the map and the multiple precision arithmetic used.
- Send me the preprint.