Universality in Dynamical Systems

Numerical Analysis of Functional Equations

How to handle Discretization Errors

Peter Wittwer
Université de Genève

Numerical Analysis of Functional Equations

How to handle Discretization Errors

What Kind of Equations?

Any equation of the form

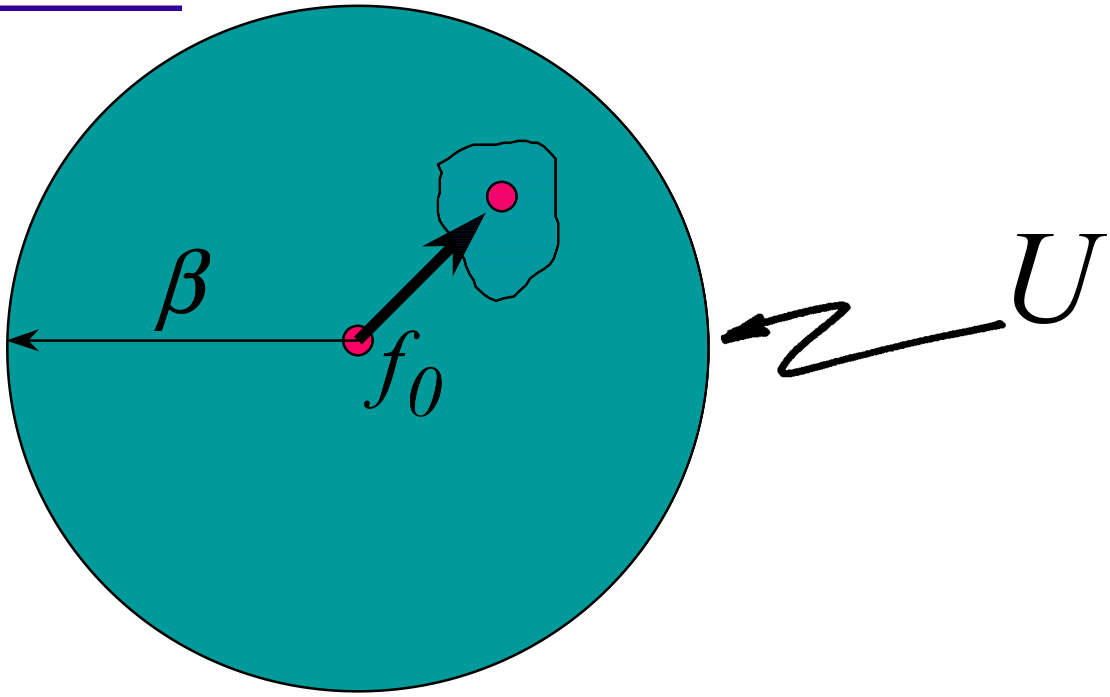
$$N(f) = f$$

$f \in B$ Banach space

$DN(f)$ reasonable

C.M.P. can be put into practice near the solution

C.M.P.



$$\| DM(f)h \| \le \rho < 1$$

where $\{ f \in U \}$ and $\{ h \in B \mid \|h\| \le 1 \}$

$$\|M(f_0) - f_0\| \le \varepsilon < (1-\rho)\beta$$

Compute with Sets

How to Proceed

- f_0 approximate solution

- $$M(f) = f - L(N(f) - f)$$

$$L \approx (DN(f_0) - 1)^{-1}$$

$$DM(f) \approx 0$$

- hypothesis of the C.M.P. satisfied

“Building Blocks”

- Decompose

$$M = M_1 \circ M_2 \cdots$$

- Bound “Factors”

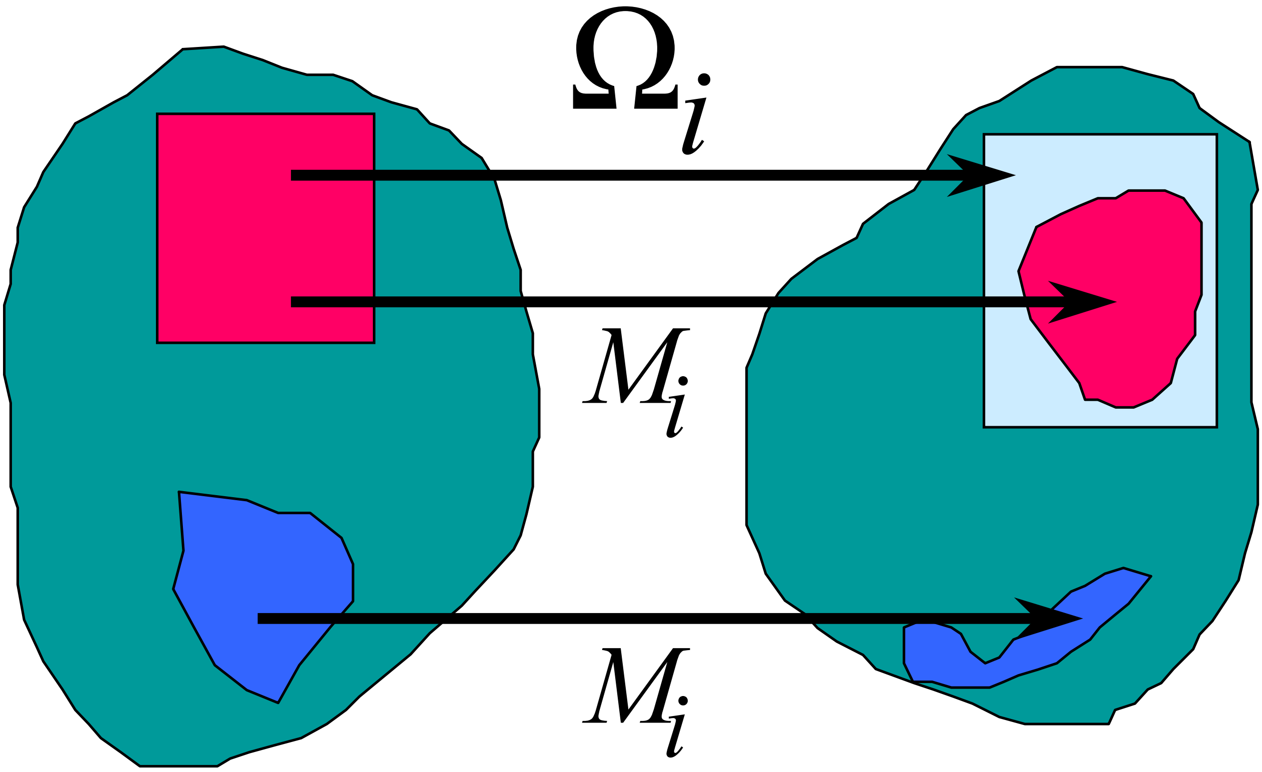
$$M_i, \quad DM_i$$

- Reassemble

$$M = M_1 \circ M_2 \cdots$$

$$DM = \text{apply chain rule}$$

Bound = Set-Map



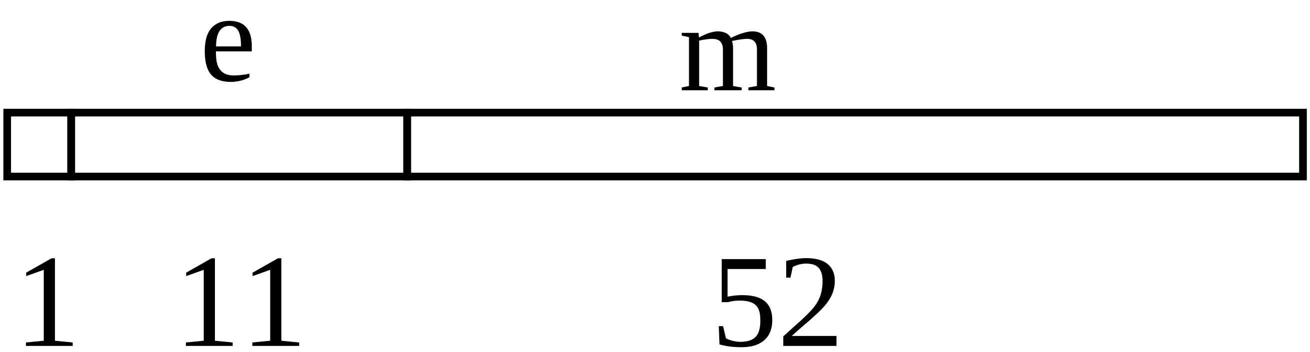
Def.: $\Omega_i \geq M_i$

$$D(\Omega_i) \subseteq D(M_i)$$

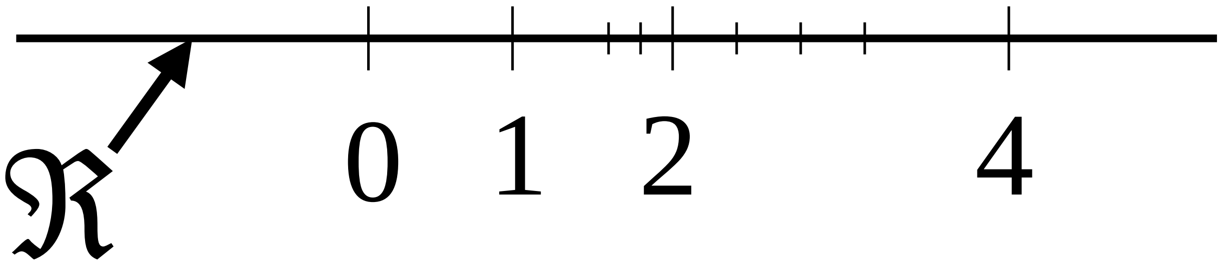
- $\Omega_i(S) \supseteq M_i(S)$

$$\forall S \in D(\Omega_i)$$

Floating Point Arithmetic (IEEE)



$$r = \pm 1.\mathrm{m}\; 2^{\mathrm{e}-1023}$$



$$r_1 \oplus_C r_2 = \mathrm{rounding}(r_1 \oplus r_2)$$

Standard Sets for **R**

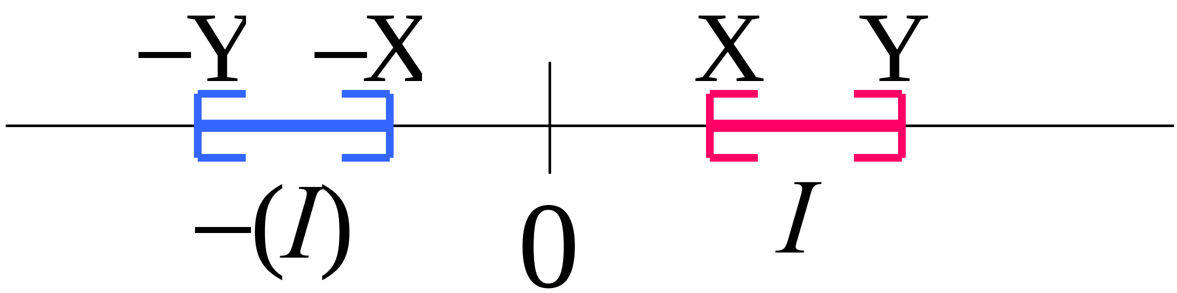
std(R) is the collection of all closed intervals i(X,Y) of the form

i(X,Y)={r∈**R**| X≤ r ≤ Y}

with X ≤ Y elements of $\Re$

Example of a Bound

$$-(I) = \{r \in \mathbb{R} \mid r = -x \text{ for some } x \in I\}.$$




```
i(X1,Y1) contains -i(X,Y):-
 X1 is -Y,
 Y1 is -X,
 !.
```

Prolog

Bounds on + and *

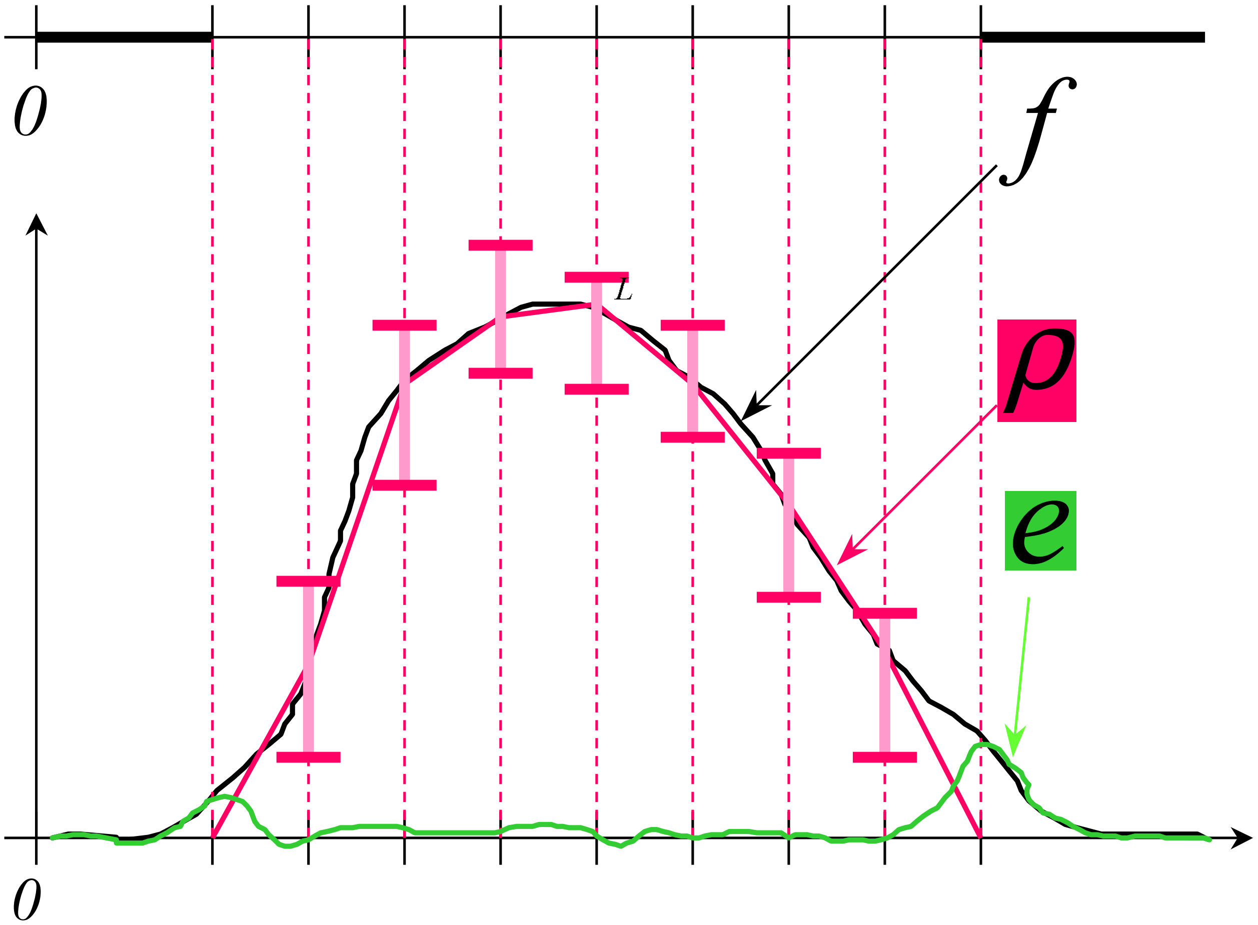
```
i(X1,Y1) contains i(X2,Y2)+i(X3,Y3):-
 X is X2+X3,
 Y is Y2+Y3,
 i(X1,Y1) enlarges i(X,Y),
 !.
```

```
i(X1,Y1) contains i(X2,Y2)*i(X3,Y3):-
 A is X2*X3,
 B is X2*Y3,
 C is Y2*X3,
 D is Y2*Y3,
 sort_numbers([A,B,C,D],[X,_,_,Y]),
 i(X1,Y1) enlarges i(X,Y),
 !.
```

Functions Spaces

$$B = L_1(R^+, w(x)\,dx)$$

$$w(x) = \exp(1/x + x)$$



$$f = \rho + e, \quad \rho \in \cdots,$$

$$\|e\| \le \varepsilon$$

Bound on Convolution

$$p(f)(x) = (f * f)(x)$$

$$f = \rho + e$$

$$f * f = \rho * \rho + 2 * \rho * e + e * e$$

$$= \tilde{\rho} + \tilde{e}$$

Lemma: $\rho * \rho$ an explicit cubic spline.

$$\|\tilde{e}\| \leq \;\ldots$$

↑

Almost no additional work needed!

Choice of Programming Language


```
C..calculate sTGJDZF and sTDZFOGAMMA
      sT3=sDIFF(sINV(sZ),sQUOT(sPOWER(sRHOP,2),sR))
      sT3=sINV(sPOWER(sSQRT(sT3),7))
      sT5=sQUOT(sPROD(sT3,sT4)
     *   ,sPROD(sPOWER(sETA,4),sPOWER(sDIFF(sONE,sT1),3)))
      sT7=sQUOT(sQUOT(siCONST(7),sTWO),sPOWER(sSQRT(sT6),9))
      sT8=sQUOT(sPROD(sSUM(sTWO,sT1),sT7)
     *   ,sPROD(sPOWER(sETA,6),sPOWER(sDIFF(sONE,sT1),5)))
      sFAC=sPROD(sQUOT(siCONST(5),sTWO),sPROD(s3HALF,sPROD(sHALF
     *    ,sSQRT(sPI))))
      sT9=sPROD(sPROD(sPROD(sTWO,sFAC),sCTT),sSUM(sT5,sT8))
      sTGJDZF=sUPPER(sPROD(sPROD(sPOWER(sRHOP,3),sT9),sGGAMMA))
      sTDZFOGAMMA=sPROD(sTWO,sPROD(sPROD(sCTT,sFAC)
     *           ,sPROD(sQUOT(sPOWER(sRHOP,4),sPOWER(sETA,3)),sT3)))
C..calculate coefficients sJ1, sJ2 and sJ3
      sT1=sQUOT(sGGAMMA,sPOWER(sRHOP,3))
      sT1=DCMPLX(-rMAXABS(sT1),rMAXABS(sT1))
      sT2=sQUOT(sT1,sPOWER(sRHOGAMMA,2))
      sJ1H=sPROD(sFP00,sPROD(sFOUR,sT1))
      sJ1 =sSUM(sJ01,sJ1H)
      sJ2H=sPROD(sFP01,sPROD(siCONST(5),sT1))
      sJ2 =sSUM(sJ02,sJ2H)
      sJ3H=sSUM(sPROD(sFP00,sSUM(sPROD(sT1,sQUOT(siCONST(12)
     *    ,sKAPPA)),sPROD(sT2,siCONST(6)))),sPROD(sFP01
     *    ,sPROD(siCONST(6),sT1)))
      sJ3 =sSUM(sJ03,sJ3H)
C..calculate bound for general
      sT1=sSUM(sSUM(sSUM(sGDZJ,sPROD(sSUM(sPROD(sABS(sFP00),sGAMMA)
     *   ,sSUM(sPROD(sABS(sFP01),s2GAMMA),sPROD(sABS(sFP02)
     *   ,sPROD(sGAMMA,s2GAMMA))))),sGDZGAMMA))
     *   ,sSUM(sPROD(sTDZFOGAMMA,sGDZGAMMA)
     *   ,sPROD(sSUM(sABS(sFP00),sSUM(sPROD(sABS(sFP01),sSUM(sGAMMA
     *   ,sGAMMA)),sPROD(sABS(sFP02),sSUM(sPROD(sTHREE,s2GAMMA)
     *   ,sSUM(sPROD(sTHREE,sPROD(sGAMMA,sGHGAMMA))
     *   ,sPOWER(sGHGAMMA,2)))))),sPROD(sDZGAMMA,sGGAMMA))))
     *   ,sPROD(sTGJDZF,sDZGAMMA))
      sT2=sSUM(sSUM(sPROD(sJ1H,sPOWER(sRHOP,3))
     *   ,sPROD(sJ2H,sPOWER(sRHOP,4))),sPROD(sJ3H
     *   ,sPOWER(sRHOP,5)))
      sGDZJTOT=sCONST(rMAXABS(sSUM(sT1,sT2)))
C..calculate bound for higher order
      sT1=sSUM(sSUM(sHDZJ,sSUM(sPROD(sDZFOGAMMA,sHDZGAMMA)
     *   ,sPROD(sSUM(sABS(sFP00),sSUM(sPROD(sABS(sFP01)
     *   ,sSUM(sGAMMA,sGAMMA)),sPROD(sABS(sFP02),sSUM(sPROD(sTHREE
     *   ,s2GAMMA),sSUM(sPROD(sTHREE,sPROD(sGAMMA,sGHGAMMA))
     *   ,sPOWER(sGHGAMMA,2)))))),sPROD(sDZGAMMA,sHGAMMA))))
     *   ,sPROD(sQUOT(sCT,sC),sPROD(sHJDZF,sDZGAMMA)))
      sHDZJTOT=sCONST(rMAXABS(sT1))
C..invert Ecalle's equation: lower order coefficients
      sDZF01=sQUOT(sJ1,sPROD(sFOUR,sGA3))
      sDZF02=sQUOT(sJ2,sPROD(siCONST(5),sGA3))
      sDZF03=sQUOT(sDIFF(sJ3,sPROD(sSUM(sPROD(s3HALF
     *      ,sQUOT(sGA5,sGA3)),sPROD(sQUOT(sTHREE,sFOUR),sGA3))
     *      ,sJ1)),sPROD(siCONST(6),sGA3))
C..add additional contribution to general
      CALL FsDILAT(vGAMMA,sRHO1,vT1)
      CALL FsMULT(vT1,sRHOGAMMA,vT1)
      CALL FDZsDILAT(vGAMMA,sRHO1,vT2)
      CALL FMULT(vT1,vT2,vT2)
      CALL FEQUAL(vT2,vT3)
```

Prolog
(PROgramming in LOGic)

even if r_1 and r_2 are in S, as we shall now assume. However, as far as bounds are concerned, it is sufficient to find an interval `i(X1,Y1)` that contains $r_1 \# r_2$, given an interval `i(X,Y)` that contains the computed value for $r_1 \# r_2$. The following predicate `enlarges/2` serves this purpose:

```
i(X1,Y1) enlarges i(X,Y):-
 X1 is_a_safe_lower_bound_on X,
 Y1 is_a_safe_upper_bound_on Y,
 !.
```

To be more precise, if `X`$\leq$`Y` are given representable numbers (not necessarily in the safe range), then `i(X1,Y1) enlarges i(X,Y)` is satisfied if and only if the indicated safe lower bound `X1` and safe upper bound `Y1` can be found, in which case `i(X1,Y1)` is a standard set with the above-mentioned property, as explained in §3. A typical application of `enlarges/2` is given in the next clause.

Consider now the sum function from $\mathbb{R} \times \mathbb{R}$ to $\mathbb{R}$. The corresponding set map assigns to a pair (I_2, I_3) in $\mathcal{P}(\mathbb{R}) \times \mathcal{P}(\mathbb{R})$ the set

(5.3) $$I_2 + I_3 = \{r \in \mathbb{R} \mid r = x + y \text{ for some } x \in I_2 , y \in I_3\}$$

in $\mathcal{P}(\mathbb{R})$. The following clause defines a bound on this map:

```
i(X1,Y1) contains i(X2,Y2)+i(X3,Y3):-
 X is X2+X3,
 Y is Y2+Y3,
 i(X1,Y1) enlarges i(X,Y),
 !.
```

As indicated in §4, the domain of this bound is the set of all pairs (`I2`, `I3`) in std($\mathbb{R}$)$\times$std($\mathbb{R}$) for which `I1 contains I2+I3` is true. In this case, the domain is determined by the predicate `enlarges/2`, which is used in order to compensate for possible rounding errors introduced by `is/2` and to ensure that the returned result `I1` is again a standard set.

The same description applies to our bound on the product of (sets of) real numbers if $+$ is replaced by $*$. The bound itself is defined as follows:

```
i(X1,Y1) contains i(X2,Y2)*i(X3,Y3):-
 A is X2*X3,
 B is X2*Y3,
 C is Y2*X3,
 D is Y2*Y3,
 sort_numbers([A,B,C,D],[X,_,_,Y]),
 i(X1,Y1) enlarges i(X,Y),
 !.
```

Here, we have used that the product of two standard sets I_2 and I_3 is an interval and that the

Discretization errors

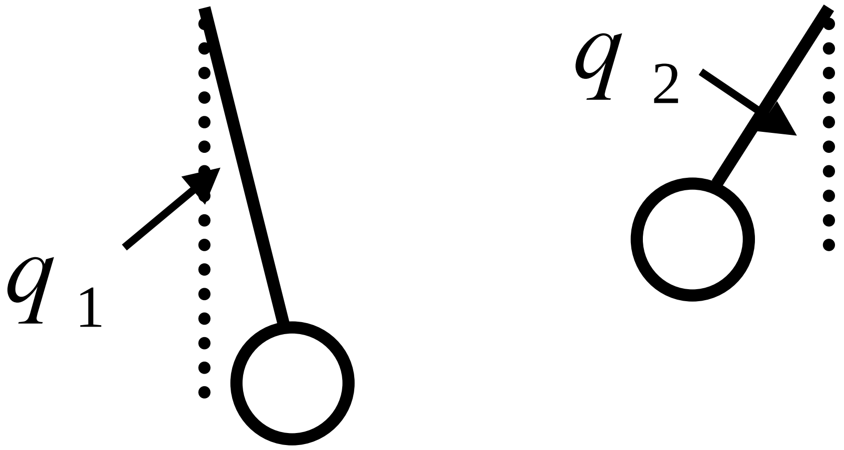
- Bounds are set-maps
- Bounds are defined on "standard sets"
- Can compute with bounds
- Bounds can be composed to form new bounds
- Can use chain rule to compute derivative of complicated maps
- Need only worry about simple maps
- Let computer do the rest
- SIAM Review, 38, 4, 1996
- Alain Schenkel 1998

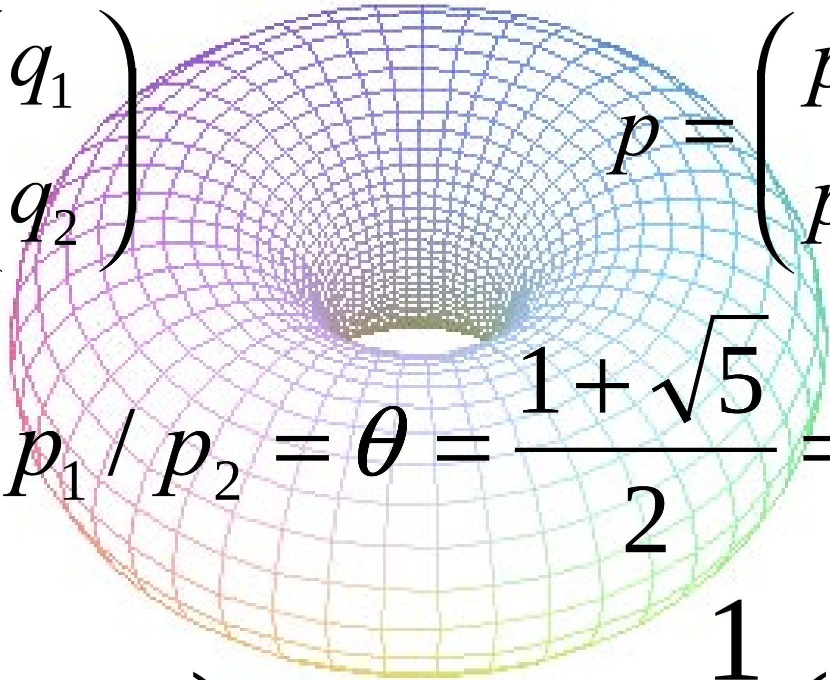
Universality in Dynamical Systems

collaboration with:

Hans Koch
Juan Abad

Hamiltonian Dynamics





$$q = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} \qquad p = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix}$$

$$\theta = p_1 / p_2 = \theta = \frac{1+\sqrt{5}}{2} = 1.61\ldots$$

$$H_0(q,p) = \omega \cdot p + \frac{1}{2}(\Omega \cdot p)^2$$

$$\partial_t p = \nabla_1 H_0 = 0$$

$$\partial_t q = -\nabla_2 H_0 = \omega + (\Omega \cdot p)\Omega$$

Fibonacci Numbers

$$T = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

$$T\omega = \theta\omega\,, \qquad T\Omega = -\frac{1}{\theta}\Omega$$

$$F_1 = F_2 = 1\,, \quad F_{k+2} = F_{k+1} + F_k$$

$$\theta_k = \frac{F_{k+1}}{F_k}\,, \qquad \theta = \lim_{k\to\infty} \theta_k$$

θ_k - tori accumulate on θ - torus

Universality

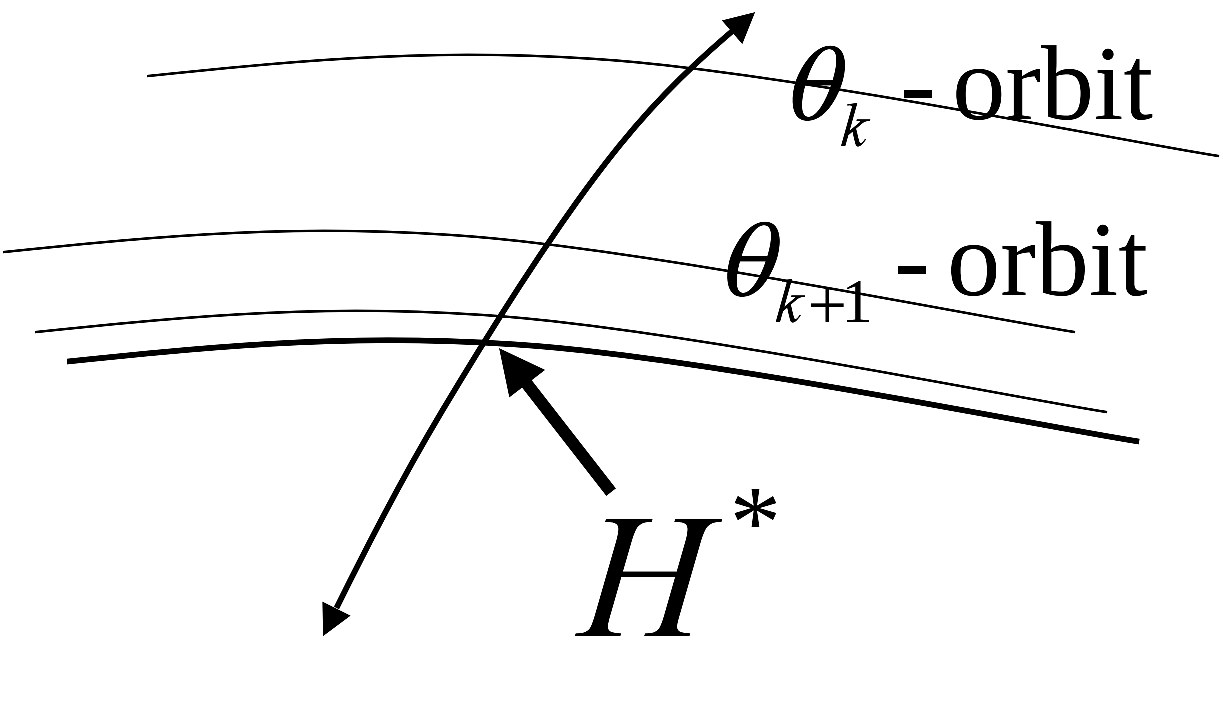
introduce coupling:

$$H = H_0 + \sum_{\substack{\nu \in Z^2 \\ n \geq 0}} h_{\nu,n} e^{i\nu \cdot q} (\Omega \cdot p)^n$$

$$N(H)(q,p) = H(Tq, T^{-1}p)$$

H has θ_k - orbit,

$N(H)$ has θ_{k-1} - orbit



Back to Computer

$$H(q,p)=\sum_{\substack{\nu\in Z^2\\ n\geq 0}} h_{\nu,n}e^{i\nu\cdot q}(\Omega\cdot p)^n$$

$$N(H)(q,p)=H(Tq,T^{-1}p)$$

ill defined, domains!

$$R(H)(q,p)=$$

$$=\frac{\tau}{\mu}H\circ U_H(Tq,\mu T^{-1}p)-\varepsilon$$

- numerical work published
- computer proof in preparation