Climbing Escher's ladder, or how to compute stability landscapes for weakly non-gradient flows

Pablo Rodríguez-Sánchez, Marten Scheffer & Egbert van Nes

2018-12-10

Abstract

Stability landscapes, usually identified with the physical concept of potential, are useful tools for understanding dynamical systems described by autonomous differential equations. Unfortunately, the conditions for those potentials to exist are quite restrictive for systems of two or more dimensions. Here we present a numerical method for decomposing dynamical systems of any size in two terms, one that has an associated potential (the gradient term), and another one that lacks it (the curl term). In those regions where the magnitude of the curl term is small compared to the gradient part, we can still make approximate use of the concept of potential. The curl to gradient ratio can be used as a local measure of the error introduced by our approximation. Both the algorithm and a ready-to-use implementation in the form of an R package are provided.

Introduction

With knowledge becoming progressively more interdisciplinary, the relevance of science communication is increasing fast. Mathematical concepts are among the hardest topics to communicate to non-expert audiences, policy makers, and even to professionals from the so-called *soft sciences*. Visual methods are known to be, when applicable, successful ways of introducing mathematical concepts and results to non-specialists.

One particularly successful visualization method is that of the stability landscape, also known as the rolling marble or ball-in-a-cup diagram (Beisner et al. [2003]). In stability landscapes (e.g.: figure 1) the horizontal position of the marble represents the state of the system at a given time. With this picture in mind, the shape of the surface represents the underlying dynamical equations, being the slope the driver of the movement. The peaks on the undulated surface represent unstable equilibrium states and the wells represent stable equilibria. Different basins of attraction are separated by *mountain ridges* in the surface. Summarizing: the marble naturally rolls downhill to the lowest point of its basin of attraction.

Stability landscapes have proven to be a successful tool to understand and explain certain non-trivial concepts about dynamical systems described by autonomous differential equations in a remarkably intuitive way. Examples of those concepts are multistability, basin of attraction and even bifurcation and hystheresis (see Beisner et al. [2003], Scheffer et al. [2001]).

The main reason for the success of this picture arises from the fact that stability landscapes are built as an analogy with our most familiar dynamical system: movement. Particularly, the movement of a marble along a curved surface under the influence of its own weight and a strong frictional force[1]. The stability landscape corresponds then with the physical concept of potential (Strogatz [1994]). This explains why our intuition, based in what we know about movement in our everyday life, works so well reading this diagrams.

Like with any other analogy, it is important to be aware of its limitations. The most important one is the fact that, for flows of more than one dimension, such a potential doesn't exist in general. To get an intutive feeling of why this is true, picture a flow with a stable cyclic attractor. We need our potential to have the shape of a surface where our marble can roll in a closed loop while always going downhill. The kind of surface that only exists on M.C. Escher's surrealistic paintings (see figure 2).

In this work we present a detailed overview of the conditions that a flow has to fulfill to be associated with a potential. Next, we center our attention in flows that fail to fulfill those conditions, and we introduce an algorithm to decompose those flows as the sum of a gradient and a curl part. Each part can be used, respectively, to compute an associated potential and to measure the local error introduced by our picture.

[1] It is important to stress the fact that under this picture there's not such a thing as inertia (Beisner et al. [2003]). The accurate analogy is that of a marble rolling in a surface while submerged inside a very viscous fluid (Strogatz [1994]).

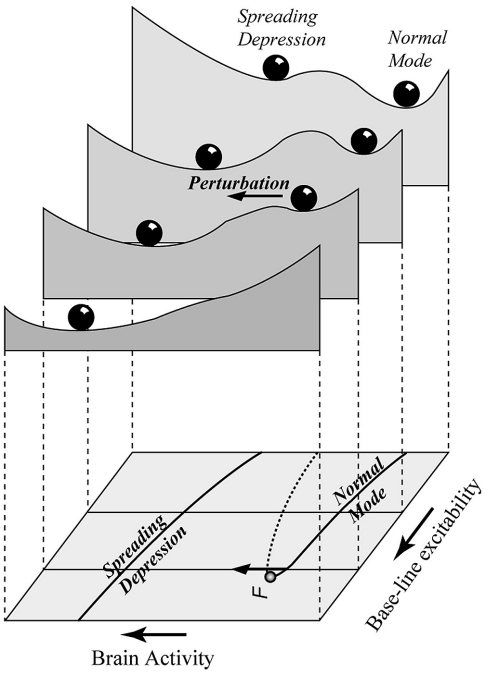

Figure 1: Example of a stability landscape from *Schefferet al.* [2001]. The upper side of the figure shows the stability landscape of a one-dimensional system for different values of a ~~bifurcation~~ parameter. The lower side shows the bifurcation diagram. This diagram proved to be a succesful tool for explaining ~~an~~ advanced concept in bifurcation theory such as the ~~F~~old bifurcation.

~~For those readers less familiar with mathematics or programming,~~ we provide a ready to use *R* package that implements the algorithm described in the present paper.

Mathematical background

~~From a mathematical point of view, the relationship between a two-dimensional scalar potential $V(x, y)$ and the dynamics it creates is given by equation 1~~

$$\begin{cases} \frac{dx}{dt} = f(x,y) = -\frac{\partial V}{\partial x} \\ \frac{dy}{dt} = g(x,y) = -\frac{\partial V}{\partial y} \end{cases} \tag{1}$$

Such a potential $V(x, y)$ exists if and only if the crossed derivatives of the ~~elements of the flow~~ are equal (equation 2). ~~Flows~~ satisfying equation 2 are known as conservative, irrotational or gradient flows (cf. section 8.3 of Marsden and Tromba [2003]).

$$\frac{\partial f}{\partial y} = \frac{\partial g}{\partial x} \tag{2}$$

If an only if condition 2 holds, we can use a line integral (Marsden and Tromba [2003], section 7.2) to invert 1 and calculate $V(x, y)$ using the ~~flow~~ equations $f(x, y)$ and $g(x, y)$ as an input. An example of this inversion is equation 3, where we have chosen an integration path composed of a horizontal and a vertical line.

$$V(x,y) = V(x_0,y_0) - \int_{x_0}^{x} f(\xi, y_0)d\xi - \int_{y_0}^{y} g(x,\eta)d\eta \tag{3}$$

The attentive reader may have raised her or his eyebr after reading the word *chosen* applied to an integration path. In fact, we can introduce this arbitrary choice without affecting the final result. The

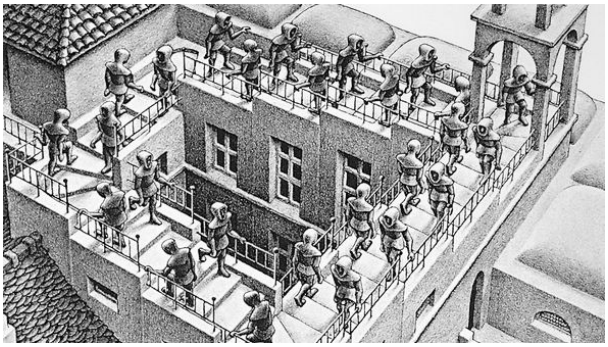

Figure 2: Detail of M.C. Escher's *Hogermen en dalen*. Lithography, 1960. In this surface, the walkers can walk in a closed loop that permanently goes downhill. Unfortunately, such a surface only exists in the imagination of ~~surrealist painters~~.

condition for potentials to exist (equation 2) is entirelly equivalent (cf. section 7.2 of Marsden and Tromba [2003]) to the path independence of any line integral between two points inside this flow. If the condition was not fulfilled, the calculated potential will have depended crucially on the chosen integration path. Being an arbitrary choice, the computed potential will have been an artifact with no natural meaning.

Generalization

Dynamics in equation 1 and the condition for the crossed derivatives 2 can be straightforwardly generalized (see equations 5 and 4) to systems with an arbitrary number of state variables $\vec{x} = (x_1, ..., x_n)$. Particularly, if and only if our flow equations $\frac{dx_i}{dt} = f_i(\vec{x})$ satisfy the condition:

$$\frac{\partial f_i}{\partial x_j} = \frac{\partial f_j}{\partial x_i} : i \neq j \tag{4}$$

then exists a potential $V(\vec{x})$ ~~verifying~~:

$$f_i(\vec{x}) = -\frac{\partial V}{\partial x_i} : i = 1..n \tag{5}$$

and such a potential can be computed using:

$$V(\vec{x}) = V(\vec{x}_0) - \int_\Gamma \sum_{i=1}^{n} f_i(\vec{x}) dx_i \tag{6}$$

where the line integral in 6 is computed along any curve Γ joining the points $\vec{x}_0$ and $\vec{x}$.

The method we'll present in the next section lies in the fact that the conditions 4 are equalities between functions, that is, both cross derivatives have to be equal at each point $\vec{x}$. Their difference, thus, will not remain constant in the whole phase space. In general, there will be regions where the conditions 4 are, at least, approximately fulfilled. Those regions will admit an approximate potential.

It is important to note that the number of equations (N) contained in condition 4 grows with the dimensionality of the system (D), particularly following the series of triangular numbers $N = \frac{1}{2}(D-1)D$. Thus, the higher the dimensionality, the harder it gets to fulfill condition 4. As a side effect, we see that in one-dimensional systems the condition contains 0 equations and is thus automatically fulfilled, meaning that one-dimensional systems always have a well defined stability landscape.

With vector notation

Using vector notation and the nabla oper. $\vec{\nabla}$ both the dynamical equation and the condition can be written compactly for any number of dimensions. Particularly, if and only if our flow equations $\frac{d\vec{x}}{dt} = \vec{f}(\vec{x})$ satisfy the condition:

$$\vec{\nabla} \times \vec{f}(\vec{x}) = \vec{0} \tag{7}$$

then ~~exists~~ a potential $V(\vec{x})$ ~~verifying~~:

$$\frac{d\vec{x}}{dt} = \vec{f}(\vec{x}) = -\vec{\nabla}V \tag{8}$$

and such a potential can be computed using:

$$V(\vec{x}) = V(\vec{x}_0) - \int_\Gamma \vec{f}(\vec{x}) \cdot d\vec{x} \tag{9}$$

The nabla operator and the line integral are used to generalize, respectivelly, the key concepts of derivative and definite integral to functions with more than one argument. For more in ation see Marsden and Tromba [2003] or any other introductory text about vector calculus.

Methods

The method we propose is based on the decomposition of a ~~flow~~ in a conservative or gradient part and a non-conservative or curl part (see equation 10).

$$\vec{f}(\vec{x}) = \vec{f}_{grad}(\vec{x}) + \vec{f}_{curl}(\vec{x}) \tag{10}$$

$\vec{f}_{grad}(\vec{x})$ captures ~~all~~ the part of the system that can be associated to a ~~scalar potential~~, while $\vec{f}_{curl}(\vec{x})$ represents the deviation from this ideal case. We'll use $\vec{f}_{gradient}(\vec{x})$ to compute ~~a scala~~ otential. The absolute error of this approach will be given by $\| \vec{f}_{curl}(\vec{x}) \|$. In those regions where the gradient term is stronger than the curl term, the previously calculated potential will represent an acceptable approximation of the underlying dynamics. The next steps show an easy way of achieving such a decomposition.

~~The reader is probably familiar with the concept of linearization,~~ also known as Taylor expansion. Any sufficiently smooth and continuous ~~flow~~ $\vec{f}(\vec{x})$ can be approximated around ~~the~~ point $\vec{x_0}$ using equation 11, where $J(\vec{x_0})$ is the jacobian matrix evaluated at ~~the approximation point~~ $\vec{x_0}$ and $\Delta\vec{x}$ is defined as the distance to the ~~approximation~~ point, that is, $\Delta\vec{x} = \vec{x} - \vec{x_0}$, written as a column vector.

$$\vec{f}(\vec{x}) \approx \vec{f}(\vec{x_0}) + J(\vec{x_0}) \cdot \Delta\vec{x} \tag{11}$$

As usual in linearization, we have neglected the terms of order 2 and higher in equation 11. This approximation is valid ~~in the surroundings of~~ $\vec{x_0}$. For an equation like 11, the condition 4 becomes a very simple restriction on the jacobian matrix: for the system to be gradient, its jacobian has to be symmetric (see equation $12, where t represents transposition$).

$$J = J^t \tag{12}$$

We know from basic linear algebra that any square matrix M can be uniquely decomposed as the sum of a skew and a symmetric matrix (e equation 13).

$$\begin{cases} M_{Symm} = \frac{1}{2}(M + M^t) \\ M_{Skew} = \frac{1}{2}(M - M^t) \end{cases} \tag{13}$$

Using the skew symmetric decomposition described in equation 13, we can rewrite 11 as:

$$\vec{f}(\vec{x}) \approx \vec{f}(\vec{x_0}) + J_{Symm}(\vec{x_0}) \cdot \Delta\vec{x} + J_{Skew}(\vec{x_0}) \cdot \Delta\vec{x} \tag{14}$$

~~As we have seen from condition 12, the only term that is not gradient is the one containing J_{Skew}.~~ Equation 14 represents a natural, well-defined and operational way of writing our ~~flow~~ $\vec{f}(\vec{x})$ decomposed as in equation 10, that is, as the sum of a gradient and a non-gradient term[2] (see 15).

$$\begin{cases} \vec{f}_{grad}(\vec{x}) \approx \vec{f}(\vec{x_0}) + J_{Symm}(\vec{x_0}) \cdot \Delta\vec{x} \\ \vec{f}_{curl}(\vec{x}) \approx J_{Skew}(\vec{x_0}) \cdot \Delta\vec{x} \end{cases} \tag{15}$$

The gradient term can thus be associated to a potential $V(\vec{x})$, whose shape can be computed analytically using a line integral (see 9). ~~Its integrated expression is given in equation 16.~~

$$\boxed{V(\vec{x}) \approx V(\vec{x_0}) - \vec{f}(\vec{x_0}) \cdot \Delta\vec{x} - \frac{1}{2}\Delta\vec{x}^t \cdot J_{Symm}(\vec{x_0}) \cdot \Delta\vec{x}} \tag{16}$$

Using this potential $V(\vec{x})$, we can rewrite our dynamical system as in equation 17. Due to the approximations used, this result will only be true on the vicinity of $\vec{x_0}$.

$$\vec{f}(\vec{x}) \approx -\vec{\nabla}V + \vec{f}_{curl}(\vec{x}) \tag{17}$$

The stability landscape described in 16 not only has been derived from a linearized system, but also completely neglects the effects of the curl part of the flow. As with any other approximation we need a way to estimate its error. From equation 17 it is apparent that we can use $\vec{f}_{curl}(\vec{x})$ as an approximation of the local error introduced by our algorithm (18).

$$\boxed{err^2(\vec{x}) \approx \|\vec{f}_{curl}(\vec{x})\|^2 \sim \|J_{Skew}(\vec{x})\|^2} \tag{18}$$

Numerical algorithm

Equation 16 can be rewritten as 19, where $\vec{x_0}$ represents the approximation point, $\vec{x_1}$ represents a neighbouring point and the potential difference between both points is given by $\Delta V(\vec{x_1}, \vec{x_0}) \equiv V(\vec{x_1}) - V(\vec{x_0})$.

$$\Delta V(\vec{x_1}, \vec{x_0}) \approx \vec{f}(\vec{x_0}) \cdot (\vec{x_1} - \vec{x_0}) - \frac{1}{2}(\vec{x_1} - \vec{x_0})^t \cdot J_{Symm}(\vec{x_0}) \cdot (\vec{x_1} - \vec{x_0}) \tag{19}$$

Provided we know the value of the potential at one point $\vec{x}_{previous}$, equation 19 allows us to estimate the potential ~~at a different point $\vec{x}_{next}$~~ (cf.: equation 20).

$$V(\vec{x}_{next}) \approx V(\vec{x}_{previous}) + \Delta V(\vec{x}_{next}, \vec{x}_{previous}) \tag{20}$$

The abovementioned ideas can be applied to the numerical computation of the approximate potential over a grid of points by following this list of steps:

1. Create a grid in the phase plane $\{\vec{x}_i\}$
2. Calculate the jacobian $J(\vec{x}_i)$ in each grid point
3. Decompose each jacobian in its skew $J_{Skew}(\vec{x}_i)$ and symmetric $J_{Symm}(\vec{x}_i)$ components

[2] This decomposition is related with the Helmholtz decomposition: it is proven that, for a huge range of easy to fulfil conditions, any flow can be decomposed as $\vec{f}(\vec{x}) = -\vec{\nabla}V(\vec{x}) + \vec{\nabla} \times \vec{A}(\vec{x})$, where V and $\vec{A}$ are known as scalar and vector potentials. Both of them have a straightforward natural interpretation, and are widely used, for instance, in electromagnetism and fluid dynamics.

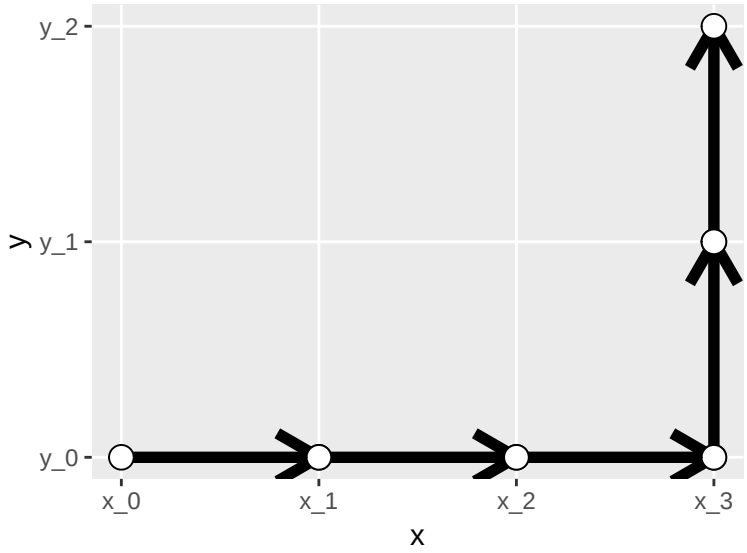

Figure 3: Path used to go from point (x_0, y_0) to (x_3, y_2). Note that this is not the only possible path. Our algorithm converges to the same potential regardless of the path chosen thanks to neglecting the skew part of the jacobian in our linearization process.

4. Assign potential 0 to the first point in the grid
5. Use equation 20 to iterativelly calculate the values of the potential at each point of the grid
6. Use the values of $J_{Skew}(\vec{x_i})$ as a measure of the quality of the approximation point by point.

By introducing a grid, we expect the linearization error to decrease with the square of the grid's step size. The more fundamental error due to ignoring the curl component of our flow cannot be affected by the grid's step choice, but the algorithm provides us with a measure of its magnitude point-by-point.

Two-dimensional example

For instance, to calculate the value of V at the point (x_3, y_2) of a grid, we should begin by assigning 0 to the potential at $V(x_0, y_0)$. Then, we need a trajectory that goes from (x_0, y_0) to (x_3, y_2), ~~stopping in~~ the intermediate grid points (see figure 3).

In the first step we go from (x_0, y_0) to (x_1, y_0). The new potential is thus (using 20):

$$V(x_1, y_0) \approx V(x_0, y_0) + \Delta V(x_1, y_0; x_0, y_0) \tag{21}$$

The next two steps continue in the horizontal direction, all the way to (x_3, y_0). The value of the potential there is:

$$V(x_3, y_0) \approx V(x_0, y_0) + \Delta V(x_1, y_0; x_0, y_0) + \Delta V(x_2, y_0; x_1, y_0) + \Delta V(x_3, y_0; x_2, y_0) \tag{22}$$

Now, to reach our destination (x_3, y_2) we have to move in the vertical direction:

$$\begin{aligned} V(x_3, y_2) \approx V(x_0, y_0) + \Delta V(x_1, y_0; x_0, y_0) + \Delta V(x_2, y_0; x_1, y_0) + \Delta V(x_3, y_0; x_2, y_0) + \\ + \Delta V(x_3, y_1; x_3, y_0) + \Delta V(x_3, y_2; x_3, y_1) \end{aligned} \tag{23}$$

Generalizing the previous example we see that, for a generic point (x_i, y_j), we can compute the approximate potential using the closed formula 24.

$$V(x_i, y_j) = V(x_0, y_0) + \sum_{k=1}^{i} \Delta V(x_k, y_0; x_{k-1}, y_0) + \sum_{l=1}^{j} \Delta V(x_i, y_l; x_i, y_{l-1}) \tag{24}$$

Formula 24 has been derived sweeping first in the horizontal direction and next in the vertical one. Of course, we can choose different paths of summation. Because we are building our potential neglecting the curl part of our flow, we know that our results will converge to the same solution regardless of the chosen path.

Code package

In the spirit of reproducible research, we published an *R* package that applies the abovementioned algorithm. It is freely available at: TBA.

Results

A synthetic example

We tested our algorithm against a flow of the form given in 25, where p_x and p_y are non-gradient perturbations. When we chose those perturbations to be zero, the flow becomes gradient and corresponds with a four-well potential. Our algorithm rendered it succesfully (cf. figure 4, left panel).

$$\begin{cases} f(x,y) = -x(x^2-1) + p_x(x,y) \\ g(x,y) = -y(y^2-1) + p_y(x,y) \end{cases} \tag{25}$$

When non-zero perturbation we introduced a four-well potential is still recognizable (figure 4, central panel). The error obtained correlates with the two perpendicular stripes of the plane (x, y) were we defined the perturbations to be maximum (figure 4, right panel).

A biological example

A simple regulatory gene network

A bistable network model can be described by a set of equations like 26 (see Bhattacharya et al. [2011]).

$$\begin{cases} \frac{dx}{dt} = b_x - r_x x + \frac{a_x}{k_x + y^n} \\ \frac{dy}{dt} = b_y - r_y y + \frac{a_y}{k_y + x^n} \end{cases} \tag{26}$$

The parameters used are identical to those in equations 6 and 7 of Bhattacharya et al. [2011], with the exception of By and $foldXY$, that we modified in order to induce an assymetry in the the dynamics (we used $By = 0.05$ and $foldXY = 1.75$).

Despite this flow is clearly non-gradient, our algorithm correctly predicts the existence of two wells (see figure 5).

Discussion

The use of stability landscapes as a helping tool to understand one-dimensional dynamical systems achieved great success, specially in interdisciplinary research communities. A generalization of the idea of scalar potential to two-dimensional systems seemed to be a logical next step. Unfortunately, as we have seen, there are subtle reasons that make two (and higher) dimensional systems fundamentally different

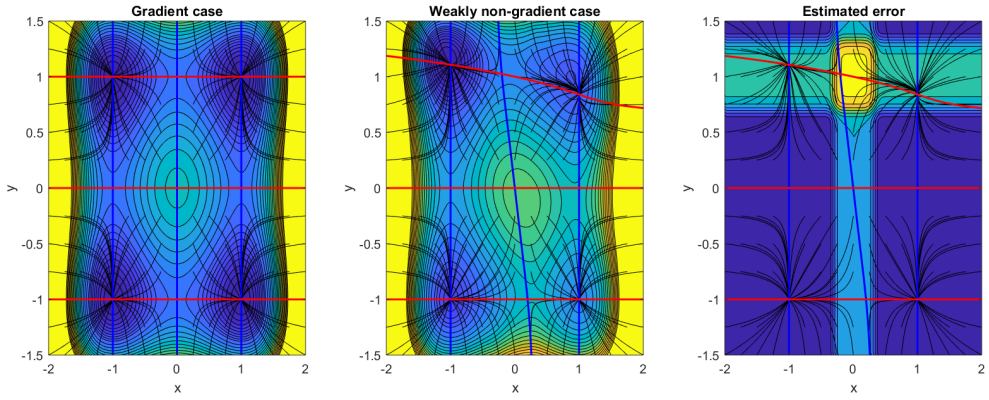

Figure 4: In all three figures the red and blue lines represent the nullclines of x and y, respectivelly. Several trajectories are plotted in black. The contour plot in figure a) shows the potential calculated for the gradient case (i.e.: $p_x(x, y) = p_y(x, y) = 0$). As expected, the stable equilibria correspond to the wells. The contour plot in figure b) also shows the potential, but now a non gradient term has been introduced. It can be noticed that the shape of the potential has been distorted. Notice also that the two equilibria at the upper side of the plot fall slightly outside their closest well. The equilibria at the bottom, to the contrary, fit perfectly centered in their corresponding wells. Figure c) shows the estimated error of our computed potential. This figure warns us against trusting the potential we obtained in the upper and central region, and guarantees us that elsewhere it will work fine. Notice that the two upper equilibria lie in a region where the error is clearly non zero. The two lower ones, on the contrary, lie on a region where the error is negligible, so we can safely use the previously derived potential to visualize them. This goes in accordance with our previous observation about the slight mismatch between equilibria and potential wells in the upper part of the figure.

from the one-dimensional case. The generalization, straightforward as it may look, is actually impossible for most dynamical equations. As a consequence, any attempt of computing stability landscapes for high-dimensional systems should, neccessarily, drop some desirable properties of classical scalar potentials.

The method proposed by Bhattacharya et al. [2011], for instance, avoids the problem of path dependence of line integrals by integrating along trajectories. The price paid is that this algorithm cannot guarantee simultaneous continuity at all equilibria and along separatrices ~~unless the underlying dynamics have a high degree of symmetry~~.

A more theoretical review of potential and pseudopotential functions computation, for a mathematically oriented audience, can be found in Zhou et al. [2012].

The algorithm we present here is an attempt to preserve as much as possible from the classical potential theory. Indeed, it calculates the closest classical potential to our problem, and the exact one when it exists. The price we pay is to calculate a potential that, in general, is only locally meaningful. The estimated error lets us know where does this happen. Additionaly, our algorithm provides:

- Simplicity. The mathematical background required is covered by any introductory course in vector calculus. Additionally, we provide the algorithm in a ready to use R package.
- ~~Reliability.~~ At each step ~~the local error is computed and stored, providing, so to say, a map of~~ which part of our stability landscape are dangerous to visit.
- Speed. The rendering of a printing quality surface can be performed in no more than a few minutes in a personal laptop.

After pointing out all the difficulties of calculating potentials for higher-dimensional systems, maybe it is worth to ask ourselves the question: do we really need them? The concept of potential is paramount in physics. In physics, potentials can be related with measurable concepts like energy, so its use goes way further than visualization. Generic stability landscapes lack this straightforward link with measurable quantities, and in most cases are just no more (and no less) than a visual aid. It may be worth reconsidering why do we prefer the idea of stability landscape over ~~the good-old~~ phase plane figure. It is true that the latter is slightly less intuitive than the stability landscape, but it has a very desirable property: existence under general circumstances.

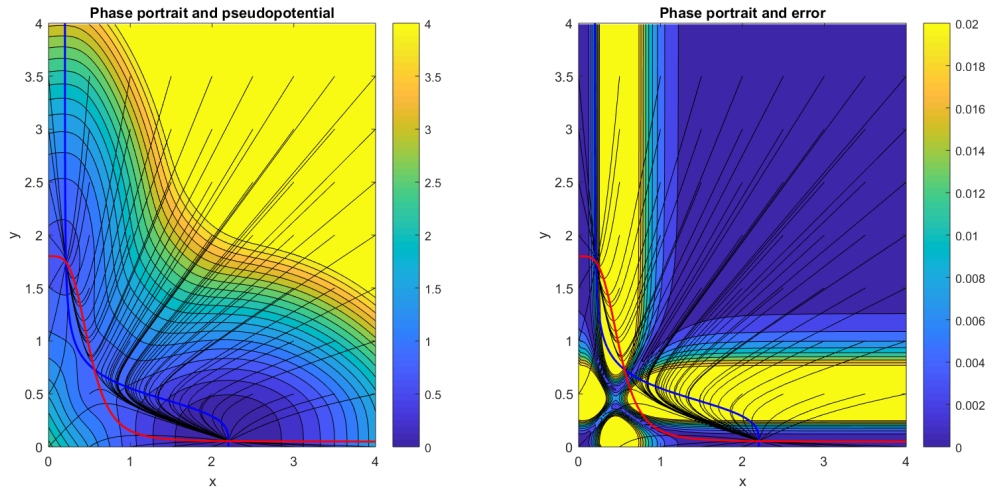

Figure 5: In both figures the red and blue lines represent the nullclines of x and y, respectivelly. Several trajectories are plotted in black. The contour plot in figure a) shows the potential calculated for the simple gene regulatory network described in equation 26. The stable equilibria correspond to the wells. Figure b) shows the estimated error of our computed potential.

Appendix

One-dimensional case

In the one-dimensional case it's always possible to find an exact scalar potential by direct integration.

$$f(x) = -\frac{dV}{dx} \implies V(x) = -\int_{x_0}^{x} f(s)ds \tag{27}$$

But, what happens if we apply our algorithm?

Around each grid point x_{n-1}, our flow will look like:

$$f(x) = f(x_{n-1}) + J(x_{n-1}) \cdot (x - x_{n-1}) : x \in [x_{n-1}, x_n) \tag{28}$$

and the potential can be approximated with:

$$V(x) = V(x_{n-1}) - f(x_{n-1}) \cdot (x - x_{n-1}) - \frac{1}{2} J(x_{n-1}) \cdot (x - x_{n-1})^2 : x \in [x_{n-1}, x_n) \tag{29}$$

After assigning $V(x_0) = V_0$ as initial value, we can use 30 to calculate the potential in any point of the grid.

$$V(x_i) = V(x_0) - \sum_{n=1}^{i} \left(f(x_{n-1}) \cdot (x_n - x_{n-1}) + \frac{1}{2} J(x_{n-1}) \cdot (x_n - x_{n-1})^2 \right) \tag{30}$$

If the grid has been constructed with a constant step Δx, equation 30 can be simplified as:

$$V(x_i) = V(x_0) - \Delta x \sum_{n=1}^{i} \left(f(x_{n-1}) + \frac{1}{2} J(x_{n-1}) \cdot \Delta x \right) \tag{31}$$

Notice that equations 30 and 31 are just algebraic relationships allowing to compute the potential at any node of the grid. Furthermore, equation 29 can be used to interpolate between the nodes.

It can be useful to define the auxiliary function:

$$\Delta V(x_n; x_{n-1}) \equiv -f(x_{n-1}) \cdot (x_n - x_{n-1}) - \frac{1}{2} J(x_{n-1}) \cdot (x_n - x_{n-1})^2 \tag{32}$$

so the recursion relation reads:

$$\boxed{V(x_i) = V(x_0) + \sum_{n=1}^{i} \Delta V(x_n; x_{n-1})} \tag{33}$$

It can be shown graphically that equation 32 approximates the analytical solution given in 27.

Some loose thoughts

About the R package

I've implemented the algorithm in Matlab, as part of a suite of visualization tools for dynamical systems I am planning to publish at some point. My plan for this paper is to translate this function alone to *R* and publish it as a package. The translation should not take much time.

Glossary

I am using as almost synonyms the terms:

- Potential (also classical potential and scalar potential)
- Stability landscape
- Surface

About notation

It will be advisable to follow a consistent notation for vector flows and its analysis in the final version of this document. Here I show a brief summary of the pros and cons of the three of them (see also accompanying figure):

Explicit notation

- Focus only on 2D systems. Provided we are focused in visualization, this is not a problem
- Understandable for readers from different backgrounds
- The mechanics of the algorithm look less clear
- Big and (arguably) ugly

Tensor notation

- Valid for systems of any size
- Understandable for readers from different backgrounds
- The matrix product appears hidden as a sum of products
- Big and (arguably) ugly

Vector notation

- Valid for systems of any size
- Less understandable for readers from different backgrounds
- The mechanics of the algorithm are written very clearly
- Pretty and compact

Why the existence of a potential fails for more than 1 dimension? Alternative approaches

From mechanics

The perpetual oscillation of a rolling marble while submerged in a very viscous fluid is impossible. This rules out the existence of potentials for systems with cyclic attractors.

Perpetual oscillation is known to be only possible in systems with two or more dimensions (by virtue of the Poincaré-Bendixson theorem (Strogatz [1994])).

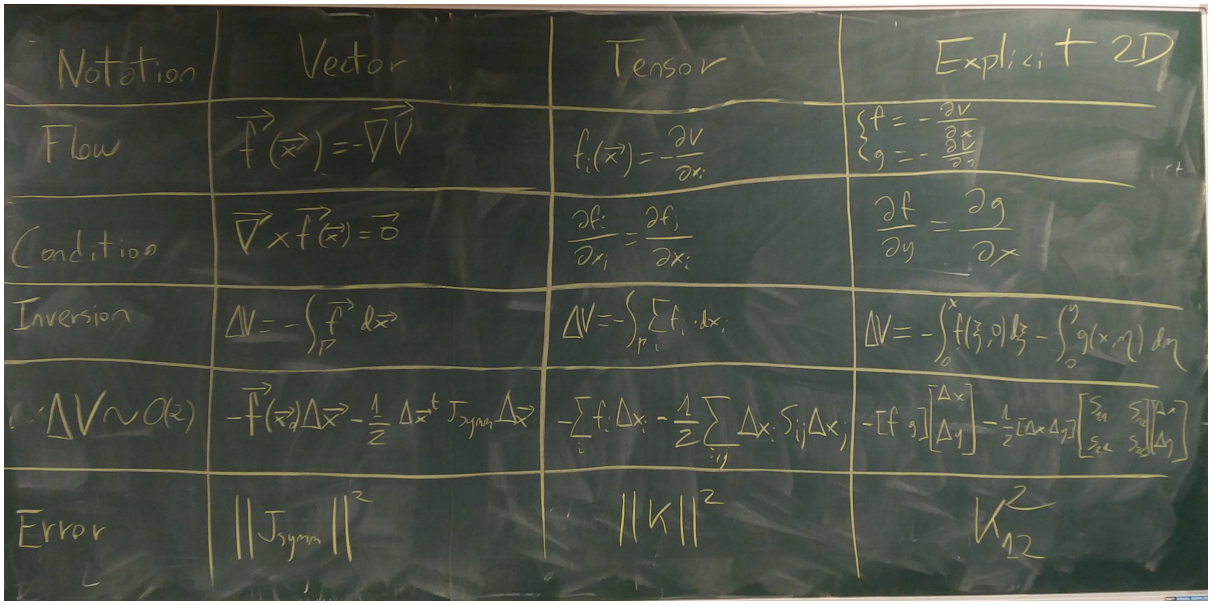

Figure 6: Summary of the three possible notations

From information theory

If we want to summarize, for instance, a two-dimensional flow in a single surface, the height at each point of such a surface should contain the information about the derivatives of both states. That is, one number, height, codes two numbers. The process of computing the surface is a process of information compression, and it is known from mathematics that only gradient flows allow this kind of compression. In one-dimensional systems there is no compression of information (one number, height, corresponds to one number, the derivative of the only state), so this restriction does not apply in the one-dimensional case.

References

BE Beisner, DT Haydon, and K. Cuddington. Alternative stable states in ecology. *Frontiers in Ecology and the Environment*, 1(7):376–382, sep 2003. ISSN 1540-9309. doi: 10.1890/1540-9295(2003)001[0376:ASSIE]2.0.CO;2. URL https://doi-org.ezproxy.library.wur.nl/10.1890/1540-9295(2003)001[0376:ASSIE]2.0.CO;2.

Sudin Bhattacharya, Qiang Zhang, and Melvin E Andersen. A deterministic map of Waddington's epigenetic landscape for cell fate specification. *BMC Systems Biology*, 5(1):85, 2011. ISSN 1752-0509. doi: 10.1186/1752-0509-5-85. URL https://doi.org/10.1186/1752-0509-5-85.

Jerrold E. Marsden and Anthony Tromba. *Vector calculus*. W.H. Freeman, 2003. ISBN 9780716749929.

Marten Scheffer, Steve Carpenter, Jonathan a Foley, Carl Folke, and Brian Walker. Catastrophic shifts in ecosystems. *Nature*, 413(6856):591–596, oct 2001. ISSN 0028-0836. doi: 10.1038/35098000. URL http://www.nature.com/doifinder/10.1038/35098000.

Steven H Strogatz. *Nonlinear Dynamics And Chaos: With Applications To Physics, Biology, Chemistry And Engineering*. 1994. ISBN 9788187169857.

Joseph Xu Zhou, M. D. S. Aliyu, Erik Aurell, and Sui Huang. Quasi-potential landscape in complex multi-stable systems. *Journal of The Royal Society Interface*, 9(77):3539–3553, dec 2012. ISSN 1742-5689. doi: 10.1098/rsif.2012.0434. URL http://rsif.royalsocietypublishing.org/cgi/doi/10.1098/rsif.2012.0434.