Physical law from experimental data

Abstract

This article proofs how to get physical laws — polynomial differential equations — starting from experimental data, with relative numerical errors. This method permits to describe every phenomenon in a compact and simple way, and chose indirectly the optimal approximation functions for a real function. It is also shortly proofed how to obtain the numerical trajectory of every differential equation

Introduction

In the last ten years there have been many works on differential equation regression. They work on noisy data from experimental data obtaining biochemical dynamic, chaotic dynamic or chaotic electronic circuit. These methods have a problem: the physical law is a geometric surface in a derivative space, therefore equal surfaces must be equal physical laws; for example $\ddot{y} - g = 0$ and $(\ddot{y} - g)^{\mathcal{M}} = 0$ have equal surfaces, solutions, and physical laws. My theory solves this dilemma using personal error function and personal F-test.

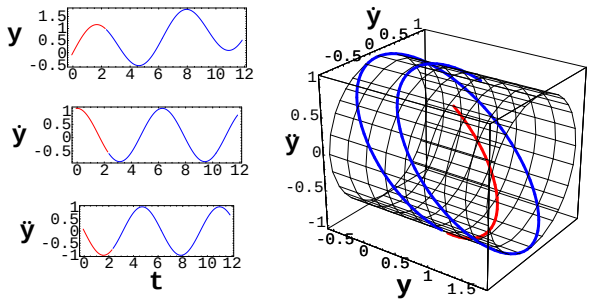

Fig. 1. Physical law and trajectory of y(t)=0.1t+sin(t)

It is clear that a surface is identified by infinite nonintersecting trajectories; it is possible to obtain the exact differential equation of a chaotic system because the trajectory is quasi-uniform on the differential equation (trajectory has noninteger dimension greater of one). I discovered that is possible to obtain the exact differential equation for non-chaotic system using a little number of non-intersecting trajectories (using different initial conditions); this can happen only for parameter-controllable systems: it is possible to use

a physical — cybernetics — model, with exact and simplified mathematical behavior of any real physical system with controlled conditions (e.g. thermodynamic, hydrodynamic, analogic electronic, etc.).

I observed with Mathematica that the differential regression trajectories are hard to obtain (numerically and analytically), then I discovered two general methods to obtain the trajectories. The first method approximates the surface in the derivative space with infinitesimal plane. The second method is similar to Parker-Sochacki method: it search an orthogonal serie minimizing the distance between trajectory and surface in the derivative space (it could be applied to celestial mechanics). I wrote open source programs to verify all these methods.

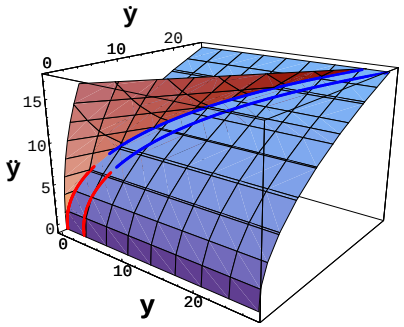

Fig. 2. Physical laws of noisy t^3 and of noisy $t^3, t^3 + 3$

Polynomial differential regression

This article proofs how to obtain the differential polynomial (physical law) in the derivative space $\{y, \dot{y}, \ddot{y}, \cdots\}$ tangent to the experimental data trajectory $\{y_{\mathrm{n}}, \dot{y}_{\mathrm{n}}, \ddot{y}_{\mathrm{n}}, \cdots\}$: some training data for the learning will be used, and the law will be verified on validation data, using a range outside the learning range.

Later on it is defined the degree of the differential equation with $\mathcal{M}$, order of the differential equation with $\mathcal{G}$ and the number of the differential equation terms with ${}^{\mathcal{M}}\mathcal{K}$; it is clear that the set of the linear differential equations is complete in the class of the ∞-times derivable function $C^{\infty}(\mathcal{R})$ (combination of Taylor, Fourier and Lagrange series).

Following I defined my differential equation error on the experimental

data (the method is in general true for mixed partial derivative):

$$
\begin{aligned}
{}^{\mathcal{M}}\mathcal{F}_{\mathrm{n}} &= \vec{w} \cdot {}^{\mathcal{M}}\vec{\mathcal{D}}_{\mathrm{n}} = w_1 + w_2 y_{\mathrm{n}} + w_3 \dot{y}_{\mathrm{n}} + w_4 \ddot{y}_{\mathrm{n}} + w_5 y_{\mathrm{n}}^2 + w_6 y_{\mathrm{n}} \dot{y}_{\mathrm{n}} + \cdots \\
{}^{\mathcal{M}}\mathcal{T}(\vec{w}) &= \sum_{\mathrm{n}} \left| \sum_{\mathrm{k}} w_{\mathrm{k}} \, {}^{\mathcal{M}}\mathcal{D}_{\mathrm{n}}^{\mathrm{k}} \right|^{1/\mathcal{M}} + \tfrac{\Lambda^2}{2} \left[1 - \sum_{\mathrm{k}} w_{\mathrm{k}}^2 \right] = \mathcal{S}(\vec{w}) + \mathcal{R}(\vec{w}) \\
{}^{\mathcal{M}}\mathcal{V}(\vec{w}) &= \sum_{\mathrm{n}} \left| \sum_{\mathrm{k}} w_{\mathrm{k}} \, {}^{\mathcal{M}}\mathcal{D}_{\mathrm{n}}^{\mathrm{k}} \right|^{1/\mathcal{M}} \\
{}^{1}\vec{\mathcal{D}}_{\mathrm{n}} &= \{1, y_{\mathrm{n}}, \dot{y}_{\mathrm{n}}, \ddot{y}_{\mathrm{n}}\} \\
{}^{\mathcal{M}}\vec{\mathcal{D}}_{\mathrm{n}} &= \left\{1, y_{\mathrm{n}}, \dot{y}_{\mathrm{n}}, \ddot{y}_{\mathrm{n}}, y_{\mathrm{n}}^2, y_{\mathrm{n}}\dot{y}_{\mathrm{n}}, y_{\mathrm{n}}\ddot{y}_{\mathrm{n}}, \dot{y}_{\mathrm{n}}^2, \dot{y}_{\mathrm{n}}\ddot{y}_{\mathrm{n}}, \ddot{y}_{\mathrm{n}}^2, \cdots, \ddot{y}_{\mathrm{n}}^{\mathcal{M}}\right\} \\
w_{\mathrm{k}} &= \frac{W_{\mathrm{k}}}{\sum_{\mathrm{k}} |W_{\mathrm{k}}|} \\
t_{\mathrm{k}} &= \begin{cases} t_0 + (k-1)\Delta^{\mathrm{T}} t & 1 \le k \le {}^{\mathrm{T}}N = N \\ t_{\mathrm{N}} + (k-N)\Delta^{\mathrm{V}} t & N < k \le {}^{\mathrm{T}}N + {}^{\mathrm{V}}N = 5N+1 \end{cases} \\
1 &\simeq \frac{4 \sum_{\mathrm{i}=1}^{\mathrm{N}-1} \sqrt{(y_{\mathrm{i}} - y_{\mathrm{i}+1})^2 + (\dot{y}_{\mathrm{i}} - \dot{y}_{\mathrm{i}+1})^2 + (\ddot{y}_{\mathrm{i}} - \ddot{y}_{\mathrm{i}+1})^2}}{\sum_{\mathrm{i}=\mathrm{N}+1}^{5\mathrm{N}} \sqrt{(y_{\mathrm{i}} - y_{\mathrm{i}+1})^2 + (\dot{y}_{\mathrm{i}} - \dot{y}_{\mathrm{i}+1})^2 + (\ddot{y}_{\mathrm{i}} - \ddot{y}_{\mathrm{i}+1})^2}}
\end{aligned}
$$

where:

- ${}^{\mathcal{M}}\mathcal{F}$ is the differential equation of degree $\mathcal{M}$
- ${}^{\mathcal{M}}\mathcal{T}(\vec{w})$ is the training error of the differential equation of degree $\mathcal{M}$
- ${}^{\mathcal{M}}\mathcal{V}(\vec{w})$ is the validation error of degree $\mathcal{M}$
- ${}^{\mathcal{M}}\mathcal{D}_{\mathrm{n}}^{\mathrm{k}}$ are the rescaled terms of the differential polynomial equation, the range of the n parameter identifies if they are validation or training terms
- w_{k} are the differential equation coefficients (the solution)
- t_{i} are the training and validation times

The error has a certain complexity to take into account some conditions:

- the training error is splitted in two part: an error $\mathcal{S}$ as the distance between differential equation and experimental data, and an regularization $\mathcal{R}$ reducing the differential equation complexity (it has attractors in points with minimum number of parameters: $W_{\mathrm{k}} = \delta_{\mathrm{ks}}$): *"Non sunt multiplicanda entia praeter necessitatem"*
- the error has normalized coefficients otherwise the solution is trivial $\vec{W} = \vec{0}$
- the coefficient normalization with the Manhattan norm — and only with this norm — has the property that every mth power of the differential equation has equal validation error
- the $\mathcal{M}$th root of the differential equation error is used because the $\mathcal{M}$th power of the differential equation has the same error. If the differential equation error on the nth data is ${}^{\mathcal{M}}\varepsilon_{\mathrm{n}} = \left(\mathcal{F}_{\mathrm{n}}\right)^{1/\mathcal{M}}$, then the pth power differential equation error is ${}^{\mathrm{p}\mathcal{M}}\varepsilon_{\mathrm{n}} = \left(\mathcal{F}_{\mathrm{n}}^{\mathrm{p}}\right)^{1/(\mathrm{p}\mathcal{M})} = \left(\mathcal{F}_{\mathrm{n}}\right)^{1/\mathcal{M}} = {}^{\mathcal{M}}\varepsilon_{\mathrm{n}}$ and the error is equal for equal surfaces

- the function and the derivatives are rescaled with $\sigma_{\ddot{y}}^2 = \sum_{\mathrm{n}} \ddot{y}_{\mathrm{n}}^2/(N_T + N_V)$ to unitary standard deviation $\ddot{z}_{\mathrm{p}} = \ddot{y}_{\mathrm{p}}/\sigma_{\ddot{y}}$ so that every term has similar values, otherwise the derivatives with low absolute value are preferred
- the measured times are chosen so to exclude the asymptotic behavior; the length of the trajectories — training and validation data — in the derivative space must be in an approximate constant ratio: the asymptote $y = k$ is a point in the derivative space

The error is modified so that the error derivative do not diverge in the points where the differential equation is null: it is used the least common divisor of the integer roots to obtain integer power of the error.

The minimum power must be greater of two, so that the diagonal part of the Hessian (necessary to minimize the validation error) is not divergent:

$$\begin{aligned}
{}^{\mathcal{M}}\mathcal{T}(\vec{w}) &= \sum_{\mathrm{n}} \left| \sum_{\mathrm{k}} w_{\mathrm{k}} \, {}^{\mathcal{M}}\mathcal{D}_{\mathrm{n}}^{\mathrm{k}} \right|^{\Phi/\mathcal{M}} + \tfrac{\Lambda^2}{2} \left[1 - \sum_{\mathrm{k}} w_{\mathrm{k}}^2 \right] \\
{}^{\mathcal{M}}\mathcal{V}(\vec{w}) &= \sum_{\mathrm{n}} \left| \sum_{\mathrm{k}} w_{\mathrm{k}} \, {}^{\mathcal{M}}\mathcal{D}_{\mathrm{n}}^{\mathrm{k}} \right|^{\Phi/\mathcal{M}} \\
\Phi &= \begin{cases} lcm(1, 2, \cdots, \mathcal{M}_{\max}) \; if \; lcm(1, 2, \cdots, \mathcal{M}_{\max}) \geq 2\mathcal{M}_{\max} \\ 2 \; lcm(1, 2, \cdots, \mathcal{M}_{\max}) \; otherwise \end{cases}
\end{aligned}$$

where the function $lcm(a, b)$ is the least common divisor of the arguments, including all the power from one to the maximum degree of the polynomial; for example the power of the polynomials of order two and degree two, three and four are $\{4, 2\}$, $\{6, 3, 2\}$, $\{12, 6, 4, 3\}$.

The discontinuity of the Manhattan norm in the error is deleted using the $\theta_{\mathrm{k}} = (W_{\mathrm{k}})^{1/3}$ coefficients (only in this way the absolute minimums are obtained)

$${}^{\mathcal{M}}\mathcal{T}(\vec{\theta}) = \sum_{\mathrm{n}} \left| \frac{\sum_{\mathrm{k}} \theta_{\mathrm{k}}^3 \, {}^{\mathcal{M}}\mathcal{D}_{\mathrm{n}}^{\mathrm{k}}}{\sum_{\mathrm{k}} |\theta_{\mathrm{k}}^3|} \right|^{\Phi/\mathcal{M}} + \frac{\Lambda^2}{2} \left[1 - \frac{\sum_{\mathrm{k}} \theta_{\mathrm{k}}^6}{\left(\sum_{\mathrm{k}} |\theta_{\mathrm{k}}^3| \right)^2} \right]$$

the gradient and the Hessians of the error are:

$$\begin{aligned}
{}^{\mathcal{M}}\Theta_{\mathrm{n}}^{\mathrm{p}} &\equiv \frac{\left| \sum_{\mathrm{k}} w_{\mathrm{k}} \, {}^{\mathcal{M}}\mathcal{D}_{\mathrm{n}}^{\mathrm{k}} \right|}{\sum_{\mathrm{k}} w_{\mathrm{k}} \, {}^{\mathcal{M}}\mathcal{D}_{\mathrm{n}}^{\mathrm{k}}} \left({}^{\mathcal{M}}\mathcal{D}_{\mathrm{n}}^{\mathrm{p}} - \frac{|w_{\mathrm{p}}|}{w_{\mathrm{p}}} \sum_{\mathrm{k}} w_{\mathrm{k}} \, {}^{\mathcal{M}}\mathcal{D}_{\mathrm{n}}^{\mathrm{k}} \right) = \frac{\left| {}^{\mathcal{M}}\mathcal{F}_{\mathrm{n}} \right|}{{}^{\mathcal{M}}\mathcal{F}_{\mathrm{n}}} \left({}^{\mathcal{M}}\mathcal{D}_{\mathrm{n}}^{\mathrm{p}} - {}^{\mathcal{M}}\mathcal{F}_{\mathrm{n}} \frac{|w_{\mathrm{p}}|}{w_{\mathrm{p}}} \right) \\
\frac{\partial \left[{}^{\mathcal{M}}\mathcal{T} \right]}{\partial W_{\mathrm{p}}} &= \frac{1}{\sum_{\mathrm{k}} |W_{\mathrm{k}}|} \left\{ \frac{\Phi}{\mathcal{M}} \sum_{\mathrm{n}} \left| {}^{\mathcal{M}}\mathcal{F}_{\mathrm{n}} \right|^{\Phi/\mathcal{M}-1} {}^{\mathcal{M}}\Theta_{\mathrm{n}}^{\mathrm{p}} - \Lambda^2 \left[w_{\mathrm{p}} - \frac{|w_{\mathrm{p}}|}{w_{\mathrm{p}}} \sum_{\mathrm{k}} w_{\mathrm{k}}^2 \right] \right\} \\
\frac{\partial^2 \left[{}^{\mathcal{M}}\mathcal{V} \right]}{\partial W_{\mathrm{p}} \partial W_{\mathrm{p}}} &= \frac{1}{\left(\sum_{\mathrm{k}} |W_{\mathrm{k}}| \right)^2} \left\{ \frac{\Phi}{\mathcal{M}} \sum_{\mathrm{n}} \left| {}^{\mathcal{M}}\mathcal{F}_{\mathrm{n}} \right|^{\Phi/\mathcal{M}-2} {}^{\mathcal{M}}\Theta_{\mathrm{n}}^{\mathrm{p}} \left[\left(\frac{\Phi}{\mathcal{M}} - 1 \right) {}^{\mathcal{M}}\Theta_{\mathrm{n}}^{\mathrm{p}} - 2 \frac{|w_{\mathrm{p}}|}{w_{\mathrm{p}}} \left| {}^{\mathcal{M}}\mathcal{F}_{\mathrm{n}} \right| \right] \right\} \\
\frac{\partial}{\partial \theta_{\mathrm{k}}} &= 3\theta_{\mathrm{k}}^2 \frac{\partial}{\partial W_{\mathrm{k}}} \\
\frac{\partial^2}{\partial \theta_{\mathrm{k}} \partial \theta_{\mathrm{k}}} &= 2\theta_{\mathrm{k}} \frac{\partial}{\partial W_{\mathrm{k}}} + 9\theta_{\mathrm{k}}^4 \frac{\partial^2}{\partial W_{\mathrm{k}} \partial W_{\mathrm{k}}}
\end{aligned}$$

the complete procedure to minimize the validation error is:

(1) initialize the differential equation coefficients in the neighborhood of the optimal solution
(2) initialize the Lagrange multiplier in the neighborhood of the optimal solution
(3) minimize the training error from the initial data
(4) in the absolute minimum point of the training error the validation error is evaluate; if the validation error is reduced (using F-test statistic), it is used as optimal solution
(5) restart from point (1)

the real differential equation is obtained reducing the validation error in a region outside the training region, measuring the generalization (extrapolation) of the differential equation.

Statistic test

I use the F-test to choose the physical law between differential equations.

The F^{Φ}-test is my nonstandard version of Fisher distribution measuring the relative variation of the validation error:

$$F^{\Phi}({}^{\mathrm{p}}\mathcal{K}, {}^{\mathrm{q}\geq\mathrm{p}}\mathcal{K}, {}^{\mathrm{V}}N) = \frac{({}^{\mathrm{p}}\mathcal{V} - {}^{\mathrm{q}}\mathcal{V})/({}^{\mathrm{q}}\mathcal{K} - {}^{\mathrm{p}}\mathcal{K})}{{}^{\mathrm{q}}\mathcal{V}/({}^{\mathrm{V}}N - {}^{\mathrm{q}}\mathcal{K})}$$

F^{Φ}-test value can be calculated only estimating numerically the sum of Φ-th power of the ε_{i} errors (they are random variables with normal density): I wrote a simple program F-test.c: it produces $25 \cdot 10^6$ random normal numbers with zero mean and unit variance used to evaluate the F^{Φ}-test values; afterwards I leave out the distribution tail with 0.1% probability (0.999 quantile); the F^{Φ}-test provides the following restriction:

$${}^{\mathrm{q}\geq\mathrm{p}}\mathcal{V} < \frac{{}^{\mathrm{p}}\mathcal{V}}{1 + ({}^{\mathrm{q}}\mathcal{K} - {}^{\mathrm{p}}\mathcal{K})\, F^{\Phi}({}^{\mathrm{p}}\mathcal{K}, {}^{\mathrm{q}}\mathcal{K}, {}^{\mathrm{V}}N)/\,({}^{\mathrm{V}}N - {}^{\mathrm{q}}\mathcal{K})}$$

Table 1
F^{Φ}-tests with 0.1% significance level. F-tests using Cochran's theorem are given in parentheses.

Φ	$F^{\Phi}(10,4,120)$	$F^{\Phi}(20,10,120)$	$F^{\Phi}(20,4,120)$
2	(4.07) 16.5	(3.30) 10.3	(2.78) 7.08
4	55.1	34.1	22.4
6	206.	128.	82.9

the validation error reduction alone is not enough to obtain the physical law, it exists a lower threshold that considers the statistical fluctuation of $\varepsilon_{\rm i}$ errors

The program

The oricchio_237D.c program minimizes the training error using the Conjugate Gradient method: it uses information about two subsequent minimizations along two lines to build a sequence of optimal search directions; if the gradient evaluation is not correct, then the search directions are not optimal: this is the reason because the directional derivative was chosen to obtain the minimum point along the line (unlike other method).

The training error is minimized along the conjugate unit vectors $\vec{n}$, rather than along the conjugate directions $\vec{p}$ (the first index is the Gradient Descent step in the line search, the second index counts the line search):

$$
\begin{aligned}
\vec{\theta}_{\rm k+1,j} &= \vec{\theta}^{\rm min} + \alpha_{\rm k+1,j}\ \vec{n}_{\rm j} \\
\alpha_{\rm k+1,j} &= \alpha_{\rm j} - v_{\rm k+1,j}\ \vec{n}_{\rm j} \cdot \nabla \mathcal{T}^{\rm min} \\
v_{\rm k+1,j} &= \begin{cases} \|\vec{\theta}^{\rm min}\|_1/(\varepsilon + |\vec{n}_{\rm j} \cdot \nabla \mathcal{T}^{\rm min}|)\ and\ \alpha_j = 0 & if\ k = 0 \\ 1.372 v_{\rm k,j}\ and\ \alpha_{\rm j} = \alpha_{\rm k,j}\ and\ \mathcal{T}^{\rm min} = \mathcal{T}(\vec{\theta}_{\rm k,j)} & if\ \mathcal{T}(\vec{\theta}_{\rm k,j}) \leq \mathcal{T}^{\rm min} \\ 0.1 v_{\rm k,j} & if\ \mathcal{T}(\vec{\theta}_{\rm k,j}) > \mathcal{T}^{\rm min} \end{cases}
\end{aligned}
$$

this condition avoid the singularity (because of the finite precision of the machine) near the minimum; the initial velocity of descent modifies the differential equation coefficients in the solution region.

The training error minimization is interrupted along the line when the training error is constant in three subsequent reductions (even if the coefficient variations are not negligible):

$$
\mathcal{T}(\vec{\theta}_{\rm k,j}) = \mathcal{T}(\vec{\theta}_{\rm s>k,j}) = \mathcal{T}(\vec{\theta}_{\rm p>s,j}) = \mathcal{T}^{\rm min}
$$

the Polak-Ribière method is used to calculate the $\beta^{\rm PR}$ parameter used to find the conjugate direction $\vec{p}_{\rm j}$:

$$
\begin{aligned}
\beta_{\rm j}^{\rm PR} &= \frac{\nabla \mathcal{T}_{\rm j+1} \cdot \left(\nabla \mathcal{T}_{\rm j+1} - \nabla \mathcal{T}_{\rm j}\right)}{\nabla \mathcal{T}_{\rm j} \cdot \nabla \mathcal{T}_{\rm j}} \\
\vec{p}_{\rm j+1} &= -\nabla \mathcal{T}_{\rm j+1} + \beta_{\rm j}^{\rm PR}\ \vec{p}_{\rm j} \\
\vec{n}_{\rm j+1} &= \frac{\vec{p}_{\rm j+1}}{\|\vec{p}_{\rm j+1}\|_1}
\end{aligned}
$$

the $\beta_{\mathrm{j}}^{\mathrm{PR}}$ parameter is set to zero when the gradients in the conjugate directions are not orthogonal (it restarts with the Gradient Descent using the Powell's restarting criterion):

$$|\nabla \mathcal{T}_{\mathrm{j}+1} \cdot \nabla \mathcal{T}_{\mathrm{j}}| > 0.1 \ \nabla \mathcal{T}_{\mathrm{j}+1} \cdot \nabla \mathcal{T}_{\mathrm{j}+1}$$

moreover $\beta_{\mathrm{j}}^{\mathrm{PR}}$ is set to zero, and the coefficients are renormalized (the gradient is reevaluated) if $\|\vec{\theta}^{\min}\|_1 < \theta_{\max}^{-1}$ or $\|\vec{\theta}^{\min}\|_1 > \theta_{\max}$, that is if the norm has not numerical representation.

The initial coefficients of the differential equation are chosen in the following way:

- an Evolutionary Strategy, or a Gradient Descent strategy (Becker-Le Cun-Ricotti) is used; it covers the neighborhood of the minimum point of validation error:

$$\begin{aligned}
\theta_{0,0}(s) &= \begin{cases} \theta^{\mathrm{opt}}(s) - \Gamma(0,\alpha_{\mathrm{m}}) \frac{\partial_{\mathrm{s}} \mathcal{V}^{\mathrm{opt}}}{\partial_{\mathrm{s}}^2 \mathcal{V}^{\mathrm{opt}}} & \textit{first strategy} \\ \theta^{\mathrm{opt}}(s) - \Gamma(0,\alpha_{\mathrm{m}}) \theta^{\mathrm{opt}}(s) & \textit{second strategy} \\ \theta^{\mathrm{opt}}(s) - \Gamma(0,\alpha_{\mathrm{m}}) \|\vec{\theta}^{\mathrm{opt}}\|_1 & \textit{from third to sixth strategy} \end{cases} \\
\Gamma(0,\alpha_{\mathrm{m}}) &= \textit{gaussian distribution variable with mean } 0 \textit{ and variance } \alpha_{\mathrm{m}} \\
\alpha_{\mathrm{m}+1} &= \begin{cases} \alpha_{\mathrm{m}} \ 1.1 \textit{ and } \vec{\theta}^{\mathrm{opt}} = \vec{\theta}^{\min} & \textit{if } \mathcal{V}^{\min} < \mathcal{V}^{\mathrm{opt}} \\ \alpha_{\mathrm{m}}/1.1 & \textit{if } \mathcal{V}^{\min} \geq \mathcal{V}^{\mathrm{opt}} \textit{ and } \alpha_{\mathrm{m}} > 10^{-3} \\ 1.0 \textit{ and change strategy otherwise} \end{cases}
\end{aligned}$$

- an Evolutionary Strategy on Lagrange multiplier is used:

$$\begin{aligned}
\Lambda_{\mathrm{m}+1} &= \begin{cases} \mathcal{S}^{1/2}[1 + \Gamma(0,\alpha_{\mathrm{m}})] \ \textit{if } \Upsilon(0,1) \leq 0.5 \\ \Lambda_{\mathrm{m}}[1 + \Gamma(0,\alpha_{\mathrm{m}})] \quad \textit{otherwise} \end{cases} \\
\Upsilon(0,1) &= \textit{uniform distribution variable in the interval } [0,1]
\end{aligned}$$

  the regularization must be of the same order of the training error magnitude, otherwise it can be numerically negligible (if the initial value is wrong or if a drastic reduction of the training error occurs): a quick alignment method is used

other conditions are listed below:

- the minimization is stopped, and the validation error is multiplied by 10, if the number of training error evaluation is more of 10^5
- the minimization is stopped, and the validation error is evaluated, if the relative reduction of the training error in $4\mathcal{K}$ line search is less than 10^{-5}

Table 2
Differential equations with relative noise 0 and 10^{-4}. The error power series $\{6, 3\}$ is used. The execution times t are in minutes and $[\mathrm{t_{min}}, \mathrm{t_{max}}]$ is the ranges of experimental times. The training data number is 30, the validation data number is 120. The number of validation evaluation for each degree is 5000.

$y(t)$	$t(0)$	$\mathcal{V}(0)$	$\mathcal{F}(0)$
$[\mathrm{t_{min}}, \mathrm{t_{max}}]$	$t(10^{-4})$	$\mathcal{V}(10^{-4})$	$\mathcal{F}(10^{-4})$
t	0.03	0.0	$0 = 1.0000\ddot{y}$
$[0, 10]$	0.02	0.0	$0 = 1.0000\ddot{y}$
te^t	93.87	$1.1 \cdot 10^{-93}$	$0 = 0.2500y - 0.5000\dot{y} + 0.2500\ddot{y}$
$[0, 10]$	94.03	$7.0 \cdot 10^{-20}$	$0 = 0.2469y - 0.4996\dot{y} + 0.2520\ddot{y} + 0.0014$
$e^{0.1t}\sin(t)$	57.47	$7.0 \cdot 10^{-95}$	$0 = 0.4570y - 0.0905\dot{y} + 0.4525\ddot{y}$
$[0, 12]$	79.92	$5.2 \cdot 10^{-22}$	$0 = 0.4570y - 0.0905\dot{y} + 0.4525\ddot{y}$
$\sin(t)$	56.52	0.0	$0 = 0.5000y + 0.5000\ddot{y}$
$[0, 12]$	67.18	$2.8 \cdot 10^{-22}$	$0 = 0.5000y + 0.5000\ddot{y}$
$\cosh(t)$	63.70	0.0	$0 = 0.5000y - 0.5000\ddot{y}$
$[0, 6]$	83.60	$3.1 \cdot 10^{-21}$	$0 = 0.5000y - 0.5000\ddot{y}$
$0.5t^2$	3.58	0.0	$0 = 0.5000 - 0.5000\ddot{y}$
$[0, 10]$	72.13	$2.0 \cdot 10^{-23}$	$0 = 0.5000 - 0.5000\ddot{y}$
$\ln(t)$	67.53	$9.3 \cdot 10^{-48}$	$0 = 0.5000\ddot{y} + 0.5000\dot{y}^2$
$[1, 10]$	33.67	$2.7 \cdot 10^{-11}$	$0 = 0.5000\ddot{y} + 0.5000\dot{y}^2$
$\ln\left(e^{2t} + 1\right) - t$	61.82	$1.9 \cdot 10^{-46}$	$0 = 0.3333 - 0.3333\ddot{y} - 0.3333\dot{y}^2$
$[0, 2]$	70.88	$7.4 \cdot 10^{-11}$	$0 = 0.3332 - 0.3334\ddot{y} - 0.3332\dot{y}^2 + 0.0002y\ddot{y}$
$\sqrt{3 + 2t}$	68.90	$3.3 \cdot 10^{-48}$	$0 = 0.5000 - 0.5000y\dot{y}$
$[-1, 10]$	38.68	$6.3 \cdot 10^{-12}$	$0 = 0.4999\dot{y}^2 + 0.5000y\ddot{y}$
$\ln(\cosh(t))$	61.38	$1.1 \cdot 10^{-47}$	$0 = 0.3333 - 0.3333\ddot{y} - 0.3333\dot{y}^2$
$[0, 3]$	54.58	$9.6 \cdot 10^{-11}$	$0 = 0.3333 - 0.3333\ddot{y} - 0.3333\dot{y}^2$
$\tan(t)$	53.18	$2.4 \cdot 10^{-47}$	$0 = 0.3333\ddot{y} - 0.6667y\dot{y}$
$[-1, 0]$	38.20	$5.1 \cdot 10^{-11}$	$0 = 0.3333\ddot{y} - 0.6667y\dot{y}$
$2\sqrt{e^t + 2}$	65.77	$7.0 \cdot 10^{-47}$	$0 = 0.3333y\dot{y} - 0.3333\dot{y}^2 - 0.3333y\ddot{y}$
$[-3, 3]$	62.78	$6.2 \cdot 10^{-11}$	$0 = 0.3317y\dot{y} - 0.3325\dot{y}^2 - 0.3311y\ddot{y} + 0.0011\dot{y} - 0.0029\ddot{y} + 0.0006\ddot{y}^2$
$0.1t + \sin(t)$	66.27	$1.1 \cdot 10^{-46}$	$0 = 0.3103 + 0.0627\dot{y} - 0.3135\dot{y}^2 - 0.3135\ddot{y}^2$
$[0, 12]$	56.78	$1.1 \cdot 10^{-10}$	$0 = 0.3103 + 0.0627\dot{y} - 0.3135\dot{y}^2 - 0.3135\ddot{y}^2$
$\sin(t) + \sin(3t)$	49.02	$2.8 \cdot 10^{-46}$	$0 = 0.4444 - 0.4375y^2 - 0.0278\dot{y}^2 - 0.0833y\ddot{y} - 0.0069\ddot{y}^2$
$[0, 12]$	37.45	$1.2 \cdot 10^{-11}$	$0 = 0.4444 - 0.4375y^2 - 0.0278\dot{y}^2 - 0.0833y\ddot{y} - 0.0069\ddot{y}^2$
$\arctan(t)$	80.67	$9.1 \cdot 10^{-09}$	$0 = 0.3290 - 0.2453y - 0.3280\dot{y} - 0.0977\ddot{y}$
$[0, 5]$	90.80	$4.7 \cdot 10^{-09}$	$0 = 0.3321 - 0.2462y - 0.3305\dot{y} - 0.0912\ddot{y}$
t^3	60.98	$8.8 \cdot 10^{-47}$	$0 = 0.9474y - 0.0526\dot{y}\ddot{y}$
$[0, 3]$	56.52	$3.3 \cdot 10^{-10}$	$0 = \boxed{0.8003y - 0.0444\dot{y}\ddot{y}} + \boxed{0.1433\dot{y} - 0.0119\ddot{y}^2}$
$\cos(t)^{-2}$	68.93	$3.3 \cdot 10^{-46}$	$0 = 0.4444y^2 + 0.3333\dot{y}^2 - 0.2222y\ddot{y}$
$[0, 1.5]$	73.77	$4.3 \cdot 10^{-10}$	$0 = \boxed{0.3649y + 0.0883\ddot{y} - 0.5294y^2} - 0.0149 - 0.0024\dot{y}$
t^{-1}	56.03	$1.6 \cdot 10^{-47}$	$0 = 0.6667\dot{y}^2 - 0.3333y\ddot{y}$
$[1, 10]$	76.55	$1.1 \cdot 10^{-12}$	$0 = \boxed{0.2170\ddot{y} + 0.4340y\dot{y}} + \boxed{0.2325\dot{y}^2 - 0.1164y\ddot{y}} - 0.0001\dot{y}\ddot{y}$
$\sin(t) + \cos(2t)$	70.68	$2.9 \cdot 10^{-40}$	$0 = 0.3683 - 0.3691y - 0.0880\ddot{y} - 0.0916y^2 - 0.0703\dot{y}^2 + 0.0010y\ddot{y} - 0.0116\ddot{y}^2$
$[0, 12]$	43.03	$2.2 \cdot 10^{-11}$	$0 = 0.2478 - 0.1104y - 0.0001\ddot{y} - 0.4033y^2 - 0.0276\dot{y}^2 - 0.1832y\ddot{y} - 0.0275\ddot{y}^2$
$\sin(t) + e^{0.1t}$	67.38	$4.2 \cdot 10^{-39}$	$0 = 0.2945 - 0.0029y^2 + 0.0583y\dot{y} - 0.2945\dot{y}^2 + 0.0583\dot{y}\ddot{y} - 0.2915\ddot{y}^2$
$[0, 12]$	78.05	$1.1 \cdot 10^{-07}$	$0 = 0.2936 - 0.0028y^2 + 0.0585y\dot{y} - 0.2941\dot{y}^2 + 0.0589\dot{y}\ddot{y} - 0.2916\ddot{y}^2 + 0.0001y - 0.0002\ddot{y}$
$e^{0.1t} + e^{0.2t}$	90.02	$3.4 \cdot 10^{-94}$	$0 = 0.0152y - 0.2273\dot{y} + 0.7576\ddot{y}$
$[0, 12]$	93.60	$8.1 \cdot 10^{-23}$	$0 = 0.0151y - 0.2272\dot{y} + 0.7577\ddot{y}$
$\exp(-t^2)$	59.52	$3.4 \cdot 10^{-47}$	$0 = 0.5000y^2 - 0.2500\dot{y}^2 + 0.2500y\ddot{y}$
$[0, 12]$	80.60	$2.0 \cdot 10^{-11}$	$0 = 0.5000y^2 - 0.2500\dot{y}^2 + 0.2500y\ddot{y}$

Results

The theory was tested with an ANSI C program, compiled with gcc version 4.1.2 20061115 with optimize level O2, on AMD Sempron(tm) 2400+, and with openSUSE 10.2 (i586).

The oricchio_237D.c program is a GNU GPL free software and it is 1056 lines long; the data are passed using another program T_F0_F1_F2.c: it creates files (training.swp and validation.swp), containing four data columns $t_\mathrm{n}, y_\mathrm{n}, \dot{y}_\mathrm{n}, \ddot{y}_\mathrm{n}$. The main program reads these two files and at the end writes four separate files:

- equa_err.swp containing the differential equation, error and gradient, for each degree
- equa_par.swp containing the physical law and gradients
- start.swp containing the initial parameters of each execution
- locked is an empty file, created at the end of the execution

The results are described in table 1 together the execution times.

Trajectory calculus

Once the physical law is obtained, it is necessary to use methods to calculate the motion trajectory (for any differential equation). The first method I discovered calculates the trajectory approximating the polynomial differential equation with the tangent plane to the trajectory point (in the derivative space), and finding the solution as Taylor's series of the linear differential equation (tangent plane) and equating the coefficient of like power of $t - t_0$:

$$
\begin{aligned}
\mathcal{F}_\mathrm{n} &= w_0 + w_1 y_\mathrm{n} + w_2 \dot{y}_\mathrm{n} + w_3 \ddot{y}_\mathrm{n} + w_4 y_\mathrm{n}^2 + w_5 y_\mathrm{n} \dot{y}_\mathrm{n} + \cdots = 0 \\
0 &\simeq \tfrac{\partial \mathcal{F}_\mathrm{n}}{\partial \ddot{y}_\mathrm{n}} (\ddot{y} - \ddot{y}_\mathrm{n}) + \tfrac{\partial \mathcal{F}_\mathrm{n}}{\partial \dot{y}_\mathrm{n}} (\dot{y} - \dot{y}_\mathrm{n}) + \tfrac{\partial \mathcal{F}_\mathrm{n}}{\partial y_\mathrm{n}} (y - y_\mathrm{n}) \\
\alpha_{\mathrm{k}\leq 2} &= \tfrac{\mathrm{d}^\mathrm{k} y(t_0)}{\mathrm{d}t^\mathrm{k}} \\
\alpha_{\mathrm{k}>2} &= -\left(\frac{\partial \mathcal{F}}{\partial \dot{y}_\mathrm{n}} \middle/ \frac{\partial \mathcal{F}}{\partial \ddot{y}_\mathrm{n}} \right) \alpha_{\mathrm{k}-1} - \left(\frac{\partial \mathcal{F}}{\partial y_\mathrm{n}} \middle/ \frac{\partial \mathcal{F}}{\partial \ddot{y}_\mathrm{n}} \right) \alpha_{\mathrm{k}-2} \\
y_{\mathrm{n}+1} &= y(t_{\mathrm{n}+1}) = y(t_\mathrm{n} + \varepsilon) \cong y(t_\mathrm{n}) + \dot{y}(t_\mathrm{n}) \varepsilon + \ddot{y}(t_\mathrm{n}) \tfrac{\varepsilon^2}{2} + \textstyle\sum_{\mathrm{k}>2} \alpha_\mathrm{k} \frac{\varepsilon^\mathrm{k}}{k!}
\end{aligned}
$$

it is not necessary to solve the differential equation to obtain the numerical solution (the same applies to differential equation with mixed derivative: membrane differential equation), and the Taylor's coefficients are in a recurrent relation (as Fibonacci sequence).

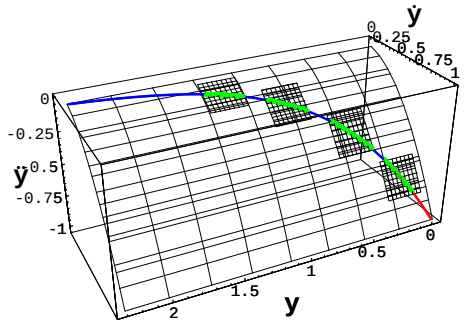

Fig. 3. Trajectory obtained approximating the surface with infinitesimal plane in the derivative space

The second method calculates the trajectory minimizing the differential equation along the trajectory:

$$\begin{aligned} V &= \tfrac{1}{2}\sum_{\mathrm{n}} \mathcal{F}_n \left(y_{\mathrm{n}}^{(0)}, y_{\mathrm{n}}^{(1)}, \cdots, y_{\mathrm{n}}^{(\mathrm{s})}\right)^2 \\ y_{\mathrm{n}}^{(\mathrm{k})} &= \sum_{\mathrm{s}} \alpha_{\mathrm{s}} \tfrac{(t_{\mathrm{n}}-t_0)^{\mathrm{s}-\mathrm{k}}}{(s-k)!} \\ \alpha_{\mathrm{s}}^{\mathrm{new}} &= \alpha_{\mathrm{s}}^{\mathrm{old}} - \eta \tfrac{\partial V}{\partial \alpha_{\mathrm{s}}^{\mathrm{old}}} = \alpha_{\mathrm{s}}^{\mathrm{old}} - \eta \sum_{\mathrm{n,k}} \left[\mathcal{F}_{\mathrm{n}} \tfrac{\partial \mathcal{F}_{\mathrm{n}}}{\partial y_{\mathrm{n}}^{(\mathrm{k})}}\right] \tfrac{(t_{\mathrm{n}}-t_0)^{\mathrm{s}-\mathrm{k}}}{(s-k)!} \end{aligned}$$

the absolute minimum is obtained with Conjugate Gradient or Broyden-Fletcher-Goldfarb-Shanno method.

Conclusion

I search the surface tangent to the experimental trajectory using polynomial regression.

If the parameter number of the polynomial regression is increased then the variance is reduced, and then the minimum solution is always the one with maximum number of parameters; a F-test statistic must be used to choose optimal polynomial evaluating the relative variation of the variance between different differential equation.

One problem is that $\ddot{y} - g = 0$ and $(\ddot{y} - g)^{\mathcal{M}} = 0$ are equal differential equations because they are equal differential surfaces; these equations must have the same error and therefore I used personal error function (depending on the degree): $\varepsilon = \frac{\ddot{y}-g}{|1|+|-g|}$ and $\varepsilon = \left[\frac{\ddot{y}^2-2g\ddot{y}+g^2}{|1|+|-2g|+|g^2|}\right]^{1/2} = \left[\frac{(\ddot{y}-g)^2}{(|1|+|-g|)^2}\right]^{1/2}$.

My personal nonstandard F-test distribution can be evaluated only numerically for random variables with normal density, using the simple included program.

The trajectory for each polynomial differential equation can be obtained numerically, and I discovered two methods: one works where Lipschitz condition is verified an the other one is always applicable.

This theory can be applied to implicit function $F\left(P,V,T\right)=0$, or $F\left(y_{\mathrm{n}},y_{\mathrm{n}-1},y_{\mathrm{n}-2}\right)$, using experimental data $\{n,P_n,V_n,T_n\}$ instead of $\{t_{\mathrm{n}},y_{\mathrm{n}},\dot{y}_{\mathrm{n}},\ddot{y}_{\mathrm{n}}\}$.

Acknowledgements

Thanks to my brother Nicola for the translations and corrections.

References

References

[1] I. Asimov, *Foundation's edge*,
Give me the idea; is a science fiction book where Hari Seldon psychohistory forecast the future

[2] L.N. Tolstoj, *War and peace*,
Tolstoj describe the Historical science using the differential of history to obtain the laws of history