UiT THE ARCTIC UNIVERSITY OF NORWAY

FACULTY OF SCIENCE AND TECHNOLOGY

DEPARTMENT OF MATHEMATICS AND STATISTICS

Tipping points and crises in financial markets

—

Polina Shemyakina

MAT-3900 Master's Thesis in Mathematics, May 2015

Abstract

Electricity spot markets and other financial markets are complex systems, and it is difficult to forecast their behaviour, especially uncontrolled and unmanageable situations, such as power crises and deflation of financial bubbles.

An energy crisis is any price rise in the supply of energy resources to an economy. It has undesirable consequences, occasionally irreversible. The most known of these crises is the California Electricity Crisis, when wholesale prices have risen by over 800%.

The goal of the thesis is to explore sudden transitions in electricity spot markets and other financial markets within the framework of nonlinear dynamical systems. There are different types of such transitions, which we can trace using observable signatures.

Using the theory of “critical slowing down”, we demonstrate that there were clear early warning signatures prior to the California Electricity Crises and other financial crises. The results are promising and show that further research should be made in this direction.

Acknowledgements

First of all, I would like to sincerely thank my supervisor, professor Martin Rypdal. I greatly appreciate his guidance and his help in working with my master thesis.

I am very grateful to other professors in Department of Mathematics and Statistics for their useful and interesting lectures. Special thanks to Kristoffer Rypdal for his course MAT-3213 “Climate Dynamics”. This course gave me knowledge, which was necessary for writing this thesis.

Finally, I thank my family and friends for the support and encouragement.

Contents

Chapter 1

Introduction

Financial markets are very complex systems, where it is hard to predict sudden fluctuations. In this thesis we review these systems as nonlinear dynamical systems.

Bifurcation theory is applied to the mathematical study of dynamical systems. It allows us to explore structural instability and critical transitions. The goal is to describe power crises and other financial crises as bifurcations in stochastic dynamical systems.

The main objectives of this thesis are:

- to review the theory of tipping points in stochastic dynamical systems;
- to explore power crises as bifurcations;
- to analyze historical electricity prices in order to look for early warning indicators.

1.1 Definition of tipping points

A tipping point is the point where a sudden transition between two states takes place. We can consider tipping points as bifurcations in stochastic dynamical system. At a tipping point, small changes in a variable can dramatically affect the state of the system at some time in the future.

The term is used in many fields, for example, ecology, climatology, and medicine. We can also describe the financial system by the general equilibrium theory and use the term of tipping point to review crises in financial markets. We focus on a set of tipping points related to financial markets.

The variance of fluctuations rises and correlation becomes longer when a tipping point is approached. It is called “critical slowing down”. It provides early warning indicators for possible crises in financial markets. The important question in financial forecasting is how to detect such indicators to avoid disastrous and unwanted events. We can find also the reasons why crises occur, and prevent similar crises from happening again.

The tipping point is the corresponding critical point of the system, at which the future state of the system is qualitatively altered. We can define control parameter p, which has a critical control value p_{crit}. The perturbation $\delta_p > 0$ causes a change $\hat{F}$ in a crucial feature of the system F [1].

Figure 1.1 shows one of the most known bifurcations, the saddle-node fold.

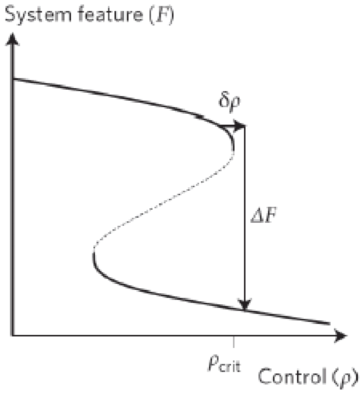


Figure 1.1: Saddle-node fold.

$|\Delta F| = |F\,(p \geq p_{\text{crit}} + \delta_p \mid T) - F\,(p_{\text{crit}} \mid T)| \geq \hat{F} > 0.$

This inequality displays that a small deviation above the critical value can cause a large qualitative change.

The point p_{crit} is a tipping point, which causes a qualitative change. The change may occur immediately, or much later.

1.2 Structure of the thesis

We describe the theory of bifurcations in dynamical systems in Chapter 2, where we give the main definitions and properties. We also review the classification of bifurcation of fixed points. The main sources of this chapter were “Nonlinear climate dynamics”, H.A. Dijkstra [2] and “Nonlinear dynamics and chaos”, S.H. Strogatz [3].

In Chapter 3 we explore bifurcations in stochastic dynamical systems, and define the basic terms in this theory. This chapter is based on lectures for the course MAT-3213 “Climate Dynamics”, professor Kristoffer Rypdal [4].

In Chapter 4 we review early-warning signals for critical transitions, and consider critical-slowing down as a leading indicator. Chapter 5 gives a description of power markets, in particular, the Californian market and its major crises.

In chapter 6 we analyze financial crises within the framework developed in this thesis and in Appendices we include computer codes used for the time series analysis.

Chapter 2

Bifurcations in dynamical systems

In this chapter we give an introduction to the theory of dynamical systems. According to the definition of a tipping point that is given in Chapter 1, we should consider bifurcations in these systems. We review bifurcations in one-dimensional systems, also in higher-dimensional systems using examples.

2.1 Dynamical systems

A dynamical system is a very important concept in mathematics. There are fixed rules that describe the time dependence of position of point in a geometrical space.

To define the dynamical system we need to determine a state space S, a set of times T and a rule R for evolution, $R : S \times T \to S$ that gives a state $s \in S$. It is a model describing the temporal evolution of a system.

These systems are used in financial and economic forecasting, environmental modelling, medical diagnosis, and other applications. Examples of dynamical systems are the model of the growth of a bacteria population and the model of an undamped pendulum.

We can consider the dynamical system as a system of first-order differential equations [2]:

$$\begin{aligned} d_t x_1(t) &= f_1(x_1, x_2, \dots, x_N) \\ d_t x_2(t) &= f_2(x_1, x_2, \dots, x_N) \\ &\dots \\ d_t x_N(t) &= f_N(x_1, x_2, \dots, x_N). \end{aligned}$$

We can write this system in the form $d_t\mathbf{x} = \mathbf{f}(\mathbf{x})$, where:

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_N \end{pmatrix}, \mathbf{f}(\mathbf{x}) = \begin{pmatrix} f_1 \\ f_2 \\ \dots \\ f_N \end{pmatrix}.$$

Here $\mathbf{f}(\mathbf{x})$ is a velocity field for the state vector $\mathbf{x}(\mathbf{t})$.

The system can be non-autonomous, if there is time dependence of velocity field $\mathbf{f}(\mathbf{x})$:

$$d_t\mathbf{x} = \mathbf{f}(\mathbf{x}, \mathbf{t}).$$

Bifurcation is a qualitative change of stability of fixed points in the dynamics, whereas bifurcation point is a parameter value at which it occurs [3].

Bifurcation is the basic concept in the mathematical study of dynamical systems. They give us models of transitions and instabilities, when one or more control parameters vary. The slight changes of parameter can cause large effect on the solution.

Bifurcations occur in one-dimensional subspace, it can also be in higher-dimensional systems.

We start with review of bifurcations of fixed points in one-dimensional systems.

Bifurcations of fixed points (for first-order equation) can be different:

- saddle-node (fold) bifurcation;

- transcritical bifurcation;
- pitchfork bifurcation.

2.2 Saddle-node bifurcation

Saddle node bifurcation (fold bifurcation) is one of the most fundamental bifurcation. It is the bifurcation when two fixed points of a dynamical system collide and destroy each other.

We often use the term of saddle-node bifurcation for continuous dynamical systems. The same bifurcation is often called fold bifurcation for discrete dynamical systems. If we change parameter, it will cause changes in the stability of a fixed point.

The simple example of saddle-node bifurcation is the first-order system:

$$\dot{x} = r + x^2,$$

where r is a bifurcation parameter, which can have different sign. This can be seen in Figure 2.1.

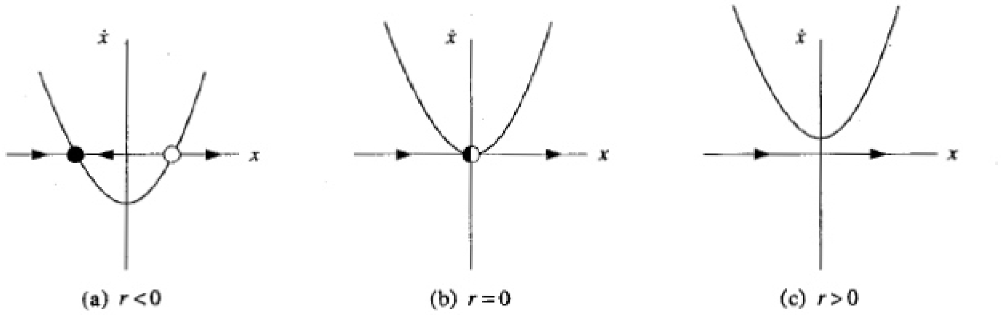


Figure 2.1: Example of saddle-node bifurcation.

There are 3 cases: parameter r is negative, positive or equal to 0. For $r \leq 0$ the system has fixed points $x = \pm\sqrt{-r}$. For $r > 0$ there are no real fixed points.

If $r < 0$ we have the stable fixed point at $x = -\sqrt{r}$ and unstable at $x = +\sqrt{r}$.

The parabola moves up when r approaches 0. In this case we have only one fixed point. Two fixed points unite into the half-stable fixed point at $x = 0$. This point is called a saddle-node fixed point.

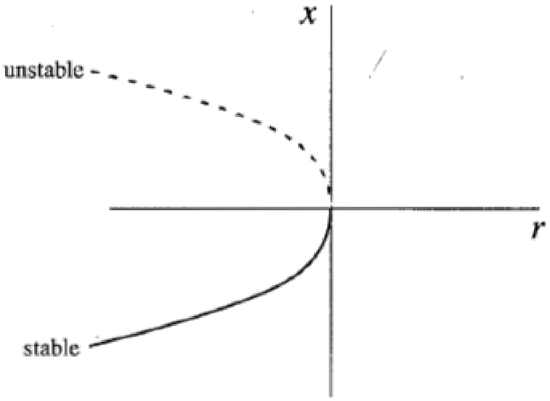


Figure 2.2: Bifurcation diagram for saddle-node bifurcation.

Figure 2.2 is the bifurcation diagram for saddle-node bifurcation. It shows how behaviour of the system changes according to the value of a bifurcation parameter r.

2.3 Transcritical bifurcation

Transcritical bifurcation is bifurcation when two fixed points exchange stability properties when a bifurcation parameter reaches a critical value.

The example for this type of bifurcation is:

$$\dot{x} = rx - x^2.$$

Figure 2.3 shows the velocity field for different value of parameter r.

The points $x = 0$ and $x = r$ are fixed points for all values of r.

In the first case, $r < 0$, we have the unstable fixed point at $x = r$ and the stable fixed point at $x = 0$.

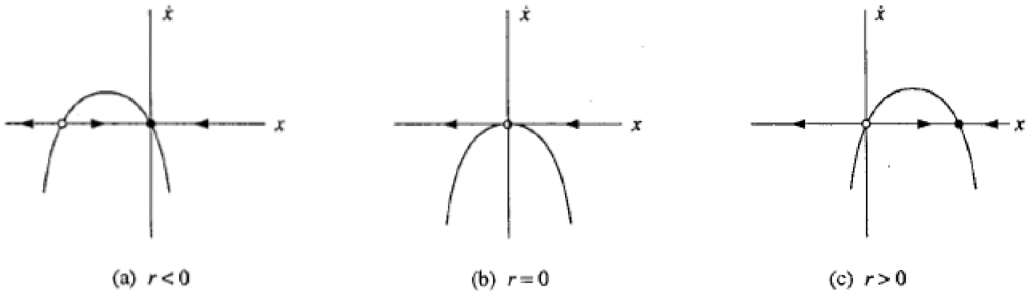


Figure 2.3: Example of transcritical bifurcation.

But for $r > 0$ the points exchange their stability, the point at $x = 0$ is unstable and the point at $x = r$ is stable. Then bifurcation occurs at $x = 0$.

There is the difference between transcritical bifurcation and saddle-node bifurcation. In transcritical bifurcation fixed points do not disappear after bifurcation, they just change their stability.

Figure 2.4 shows the bifurcation diagram for transcritical bifurcation.

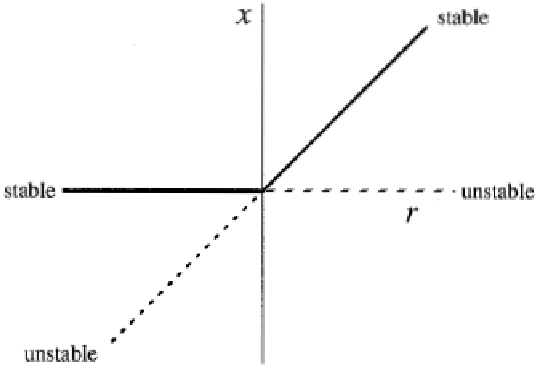


Figure 2.4: Bifurcation diagram for transcritical bifurcation.

2.4 Pitchfork bifurcation

Pitchfork bifurcation has two types: supercritical and subcritical. These bifurcations occur with symmetry.

Subcritical bifurcation takes place when an unstable branch splits into one stable and two unstable branches. Supercritical bifurcation is when one stable branch splits into one unstable and two stable branches.

2.4.1 Supercritical bifurcation

Supercritical bifurcation is given by the system:

$$\dot{x} = rx - x^3.$$

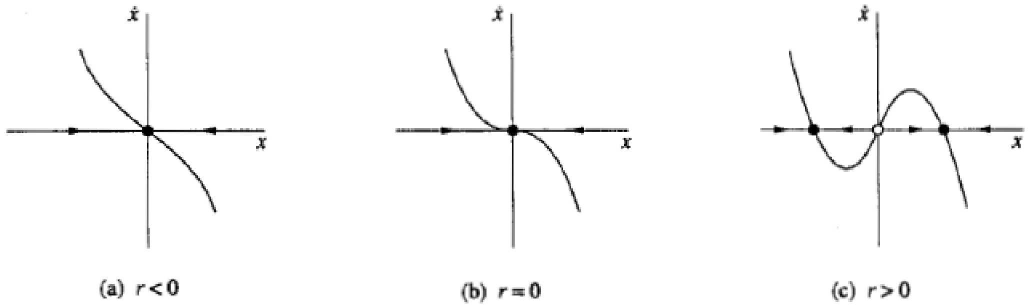


Figure 2.5: Example of Pitchfork bifurcation.

For $r < 0$ there is only one fixed point $x = 0$, and it is stable. When $r = 0$ the fixed point $x = 0$ is still stable, but weaker.

For $r > 0$ we have the fixed point $x = 0$, which is unstable, and two new stable fixed points $x = \pm\sqrt{r}$. We can see this in the Figure 2.5.

The term "pitchfork" is clearer when we plot the bifurcation diagram. It is shown in Figure 2.6(a).

The cubic term is stabilizing.

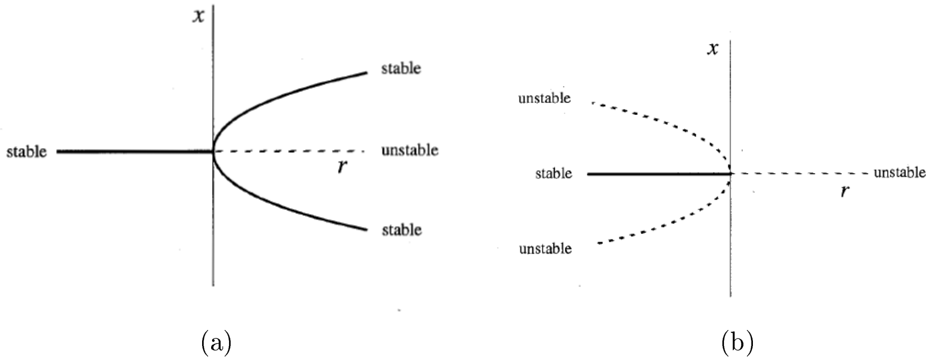


Figure 2.6: (a): Supercritical bifurcation diagram. (b): Subcritical bifurcation diagram.

2.4.2 Subcritical bifurcation

Subcritical bifurcation is given by the system:

$$\dot{x} = rx + x^3.$$

Figure 2.6(b) shows the bifurcation diagram of this system.

In this case, for $r < 0$ two nonzero fixed points $x = \pm\sqrt{r}$ are unstable, whereas the fixed point $x = 0$ is stable.

For $r > 0$ there is the fixed point $x = 0$, it is unstable.

2.5 Bifurcations in higher-dimensional systems

Bifurcations in higher-dimensional systems are analogous to bifurcations of fixed points in one-dimensional subspace. If we add more dimensions in systems, nothing really new happens. Bifurcation takes place in one dimensional subspace, and this subspace limits all the action. If we take more dimensions, the flow is either simple attraction or repulsion from that space.

The simple example of saddle-node bifurcation in two dimensions is:

$$\dot{x} = \mu - x^2, \dot{y} = -y.$$

A bifurcation behaviour in the x-direction is similar to the behaviour in the one-dimensional case. The motion is exponentially damped in the y-direction. This is illustrated in Figure 2.7.

For $\mu > 0$ we have two fixed points, one stable node at $(\sqrt{\mu}, 0)$ and a saddle at $(-\sqrt{\mu}, 0)$. As μ approaches 0, two fixed points unite and disappear for $\mu > 0$.

Even after two fixed points have eliminated each other, they continue to influence on the flow. They leave "ghost" region that absorbs trajectories and delay them before allowing passage out the other side.

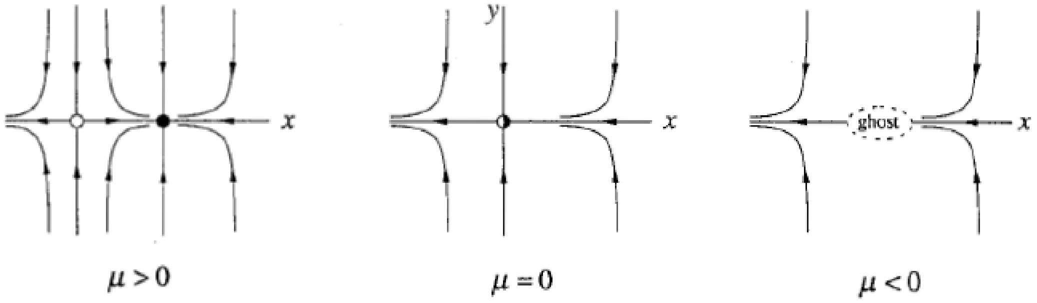


Figure 2.7: Saddle-node bifurcation in two dimensions.

We can also give the analogue of transcritical and pitchfork bifurcation in two-dimensional systems. In the x-direction it is like in the one-dimensional case, and in the y-direction dynamics is exponentially damped.

Examples of these bifurcations are:

$\dot{x} = \mu x - x^2, \dot{y} = -y$ (transcritical),

$\dot{x} = \mu x - x^3, \dot{y} = -y$ (supercritical pitchfork),

$\dot{x} = \mu x + x^3, \dot{y} = -y$ (subcritical pitchfork).

2.6 Hopf bifurcation

We have reviewed the cases in which a real eigenvalue passes trough $\lambda = 0$. Now we take the case, in which a fixed point of a dynamical system loses stability.

It occurs when complex conjugate eigenvalues of the linearization around the fixed point cross the imaginary axis of the complex plane. It is Hopf bifurcation.

The simple example is the next system:

$$d_t r = pr - r^3, d_t \theta = \omega.$$

For $p < 0$ there is one stable spiral node. It is plotted in Figure 2.8(a).

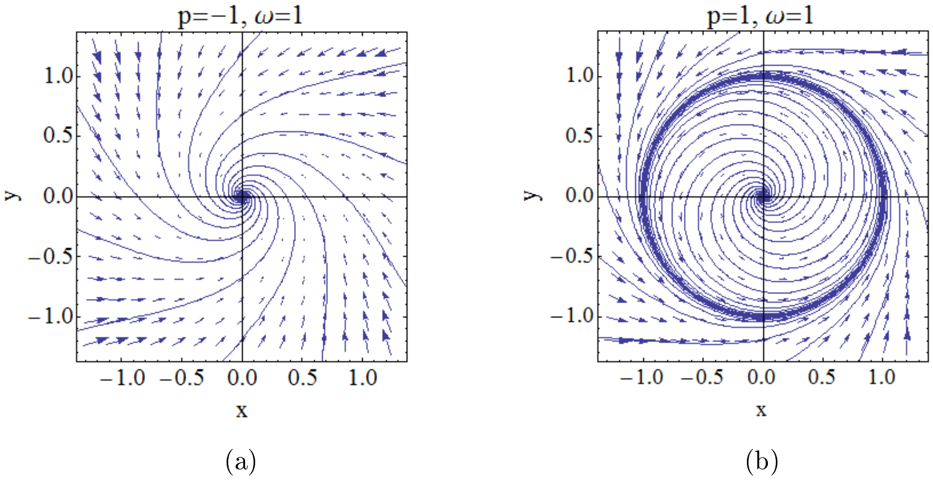


Figure 2.8: (a): Example of Hopf bifurcation for $p < 0$, stable spiral node. (b): Example of Hopf bifurcation for $p > 0$, a limit cycle.

For $p > 0$ the origin becomes an unstable spiral node, but we also have a solution with $r = \sqrt{p}$ and $\theta = \omega t$. It is a limit cycle, and since it attracts orbits both from inside and outside the circle it is stable. This can be seen in Figure 2.8(b).

Hopf bifurcation is bifurcation, when a stable spiral node becomes an unstable spiral node and a stable limit cycle.

Chapter 3

Bifurcations in stochastic dynamical systems

In this chapter we consider stochastic calculus. We define different types of stochastic processes, a stochastic integral, Ito's first lemma and a stochastic differential equation. It is the main terms in this theory. The important concept in stochastic calculus is definition of the Ornstein-Uhlenbeck process. We consider fast-slow systems. Also we describe stochastic dynamical systems and bifurcations. The Hurst exponent is the important quantitative measure of the persistence of time series. We give the algorithm of calculation of the Hurst exponent. This chapter is based mainly on lectures for the course MAT-3213 “Climate Dynamics” by professor Kristoffer Rypdal [4].

3.1 Stochastic calculus

Stochastic calculus is a branch of mathematics that studies stochastic processes. It is used to model systems that behave randomly.

Firstly, we define the basic terms in stochastic calculus.

A stochastic process X_t is a time series of random variables. It can be written in the form:

$$(X_t, t \in T) = (X_t(\omega), t \in T, \omega \in \Omega),$$

where T is a time interval, Ω is a set of elementary events. If t is fixed, $X_t(\omega)$ is just a random variable [2].

A Gaussian white-noise process is a process that has the next properties:

- it is defined in $T = [0, 1]$;
- $X_{t_1}, X_{t_2}, \ldots, X_{t_n}$ are all independent and normal distributed, where $0 \leq t_1 \leq t_2 \leq ... \leq t_n \leq 1$;
- for any $t, s \in [0, 1]$ and if $t \neq s$ we have: the expectation function $\mu_X(t) = 0$, variance $\sigma_X^2(t) = 1$, covariance $c_X(t, s) = 0$.

The most known stochastic process is a Wiener process that is used to model the Brownian motion.

The Wiener process is a process W_t, where $t \in [0, \infty]$:

- W_t has stationary, independent increments;
- $W_0 = 0$;
- W_t has continuous sample paths;
- W_t is normal distributed with $\mu_W(t) = 0$, $\sigma_W^2(t) = t$.

This process is widely applied in financial mathematics and economics. For example, it is used to model the evolution in time of stock prices.

A stochastic integral is a mean-square limit of partial sum of stochastic differentials:

$$X \equiv \int_0^T h(t) dW(t) = \mathrm{ms} - \lim_{N \to \infty} X_N \equiv \mathrm{ms} - \lim_{N \to \infty} \sum_{j=0}^{N-1} h(\tau_j)(W_{t_{j+1}} - W_{t_j}),$$

where $t_0 < t_1 < ... < t_N = T$, $t_j < \tau_j < t_{j+1}$.

The mean-square limit means:

$$X = \text{ms}-\lim_{N\to\infty} X_N \Leftrightarrow \lim_{N\to\infty} E[(X-X_N)^2] = 0.$$

The result of this integral depends on how we choose τ_j on the interval $t_j < \tau_j < t_{j+1}$. If we choose $\tau_j = t_j$, we will get the Ito integral. If we take $\tau_j = \frac{1}{2}(t_j + t_{j+1})$, it will be the Stratonovich integral.

We consider the case $\tau_j = t_j$, where $h(t) = W_t$.

Write $\Delta W_j = W_{j+1} - W_j$, we can get the partial sum:

$$X_N = \frac{1}{2}{W_t}^2 - \frac{1}{2}\sum_{j=0}^{N-1} \Delta W_j^2.$$

Then we find the Ito integral:

$$\int_0^T W_t dW(t) = \text{ms}-\lim_{N\to\infty} X_N = \text{ms}-\lim_{N\to\infty}\left(\frac{1}{2}{W_t}^2 - \frac{1}{2}\sum_{j=0}^{N-1}\Delta W_j^2\right) = \frac{1}{2}(W_t^2 - T).$$

Introduce a smooth function $f : \mathbb{R} \to \mathbb{R}$ and define $dW_x = W_{x+dx} - W_x$. The Taylor expansion is $f(W_x + dW_x) - f(W_x) = f'(W_x)dW_x + \frac{1}{2}f''(W_x)(dW_x)^2 + ...$

It gives Ito's first lemma:

$$f(W_t) - f(W_s) \overset{\text{ms}}{=} \int_s^t f'(W_x)dW_x + \frac{1}{2}\int_s^t f''(W_x)dx.$$

A first-order deterministic ordinary differential equation has the form:

$$\frac{dx}{dt} = f(x,t) \Leftrightarrow dx = f(x,t)dt.$$

We integrate this equation:

$$x(t) - x_0 = \int_0^t f(s, x(s))ds.$$

A general stochastic differential equation is:

$$dX_t = a(t, X_t)dt + b(t, X_t)dW_t,$$

where $a(t,x)$ and $b(t,x)$ are smooth functions in t and x. It has the solution:

$$X_t - X_0 = \int_0^t a(s,X_s)ds + \int_0^t b(s,X_s)dW_s.$$

The example of particular stochastic differential equation is:

$$dX_t = [(\alpha + \frac{\sigma^2}{2})X_t - \beta X_t^3]dt + \sigma X_t dW_t,$$

which solution is:

$$X_t = \frac{X_0 e^{\alpha t + \sigma W_t}}{(1 + 2X_0^2 \beta \int_0^t e^{2\alpha u + 2\sigma W_u} du)^{1/2}}.$$

The solution depends on the initial condition X_0, the time t and the sample path of the Wiener process W_u for $u \in [0,t]$.

The stochastic differential equation is linear if we write the functions $a(t,x)$ and $b(t,x)$ in the form:

$$a(x,t) = c_1(t)x + c_2(t), b(x,t) = \sigma_1(t)x + \sigma_2(t).$$

We consider the case $c_1(t) = -\gamma$, $c_2(t) = 0$, $\sigma_2 = \sigma$ (all constants). It gives the Langevin equation:

$$dX_t = -\gamma X_t + \sigma dW_t.$$

3.2 The Ornstein-Uhlenbeck process

One of the important concepts in the stochastic calculus is a definition of the Ornstein-Uhlenbeck process (OU). It was offered by Uhlenbeck and Ornstein (1930) as an alternative to Brownian motion. This process is example of the Gaussian process, that has limited variance.

The Gaussian OU process $X = (X_t)_{t \geq 0}$ can be defined as the solution to the stochastic differential equation:

$$dX_t = \gamma(m - X_t)dt + \sigma dB_t,$$

where $t > 0$, γ, m and $\sigma \geq 0$ are real constants, and B_t is standard Brownian motion. The first term of this equation on the right-hand side is the damping term (or drift term). X_0, the initial value of X, is a given random variable, taken to be independent of $B = (B_t)_{t \geq 0}$ [5]. m is the long-run mean, to which the process tends to revert, γ is the speed of mean reversion, σ measures the process volatility.

The damping term depends on the value of current state of the process. If the value of current state is more than the long-run mean, the damping term will be negative.

Variance is given by the formula:

$$\mathrm{Var}(X(t)) = \frac{\sigma^2}{2\gamma}.$$

The Ornstein-Uhlenbeck process is used in finance to model prices stochastically.

The forward Fokker-Planck equation for probability density function $p(x,t)$ for the OU process is:

$$\frac{\partial p}{\partial t} = \frac{\partial(\gamma x p)}{\partial x} + \frac{\sigma^2}{2}\frac{\partial^2 p}{\partial x^2},$$

with the boundary conditions $p, \frac{\partial p}{\partial x} \to 0$ as $x \to \infty$.

For $\frac{\partial p}{\partial t} = 0$ we get the solution:

$$p(x) = \sqrt{\frac{\gamma}{\pi}}\frac{1}{\sigma}e^{-\frac{\gamma}{\sigma^2}x^2}.$$

3.3 Fast-slow systems

Consider fast-slow systems:

$$dX_t = f(X_t, Y_t)dt + \sigma dW_t, dY_t = \varepsilon dt,$$

where $\varepsilon \ll 1$, X_t is a fast stochastic process and Y_t is a slow one.

Thr Fokker-Planck equation is $\frac{\partial p}{\partial t} = -\frac{\partial(fp)}{\partial x} + \frac{\sigma^2}{2}\frac{\partial^2 p}{\partial x^2}$.

Variable y in fast-slow systems is a bifurcation parameter, it increases slowly with time.

We consider saddle-node bifurcation, choose $f(x,y) = -y - x^2$.

We plot variance $\mathrm{Var}[X_t](y)$ using Mathematica. Variance of the stochastic process X_t increases, when y approaches the bifurcation point at $y = 0$. It is shown in Figure 3.1. The distance from the maximum variance to the bifurcation point decreases with decreasing noise level σ.

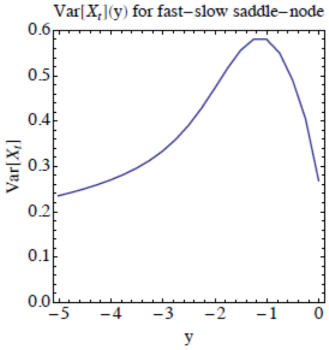


Figure 3.1: Variance for fast-slow system.

The increase of variance as y approaches bifurcation can be also found in other types bifurcations. It is an important signal of a critical transition.

The autocorrelation function is:

$$r(t) = e^{-\gamma t},$$

where the autocorrelation time also grows as the system approaches bifurcation.

3.4 Stochastic dynamical systems

A stochastic dynamical system is a solution of the set of first-order stochastic differential equations. It is a practical definition.

We can define stochastic dynamical systems mathematically. It is a mapping giving the state at the time t:

$$(t, \omega, x) \to \phi(t, \omega)[x],$$

where x is a state at $t = 0$, ω is a sample path of some continuous process satisfying $\omega(0) = 0$.

These systems are generally dynamical systems linearized around fixed points with additive white noise stochastic forcing.

They have normal form:

$$d\mathbf{x}(t) = f(\mathbf{x})dt + \sigma d\mathbf{B}(t),$$

where $d\mathbf{B}(t)$ is a n-dimensional white-noise process. $f(\mathbf{x})$ determines the deterministic part of the system, while $\sigma d\mathbf{B}(t)$ determines the stochastic part.

We return to the given example of particular stochastic differential equation. The deterministic version of this equation is:

$$dX_t = (\alpha X_t - \beta X_t^3)dt.$$

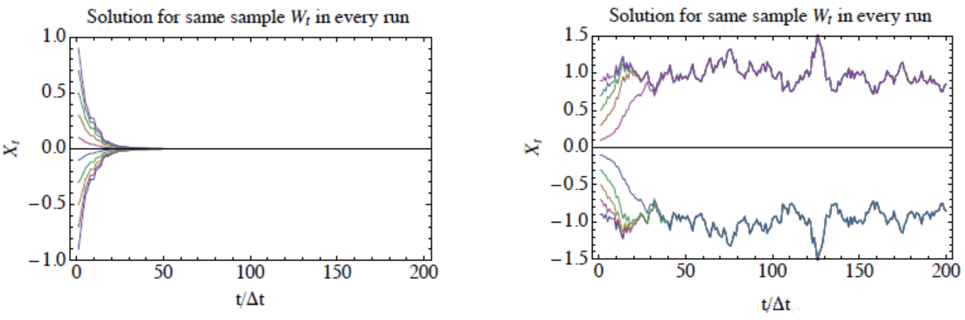
It is the equation for pitchfork bifurcation. There is one stable fixed point $x = 0$ for $\alpha < 0$. If $\alpha > 0$ this point is unstable and there are new stable fixed points $x = \pm\sqrt{\alpha}$.

We consider the stochastic case of this example. The fixed points are changed on the pullback attractors. This is illustrated in Figure 3.2. These figures show numerical solutions of the equation for different initial conditions and cases of different values α. A new sample path W_t is generated in every run.

The covariance of two random variables X and Y is:

$$\mathrm{Cov}[XY] = E[(X - \mu_X)(Y - \mu_Y)],$$

and the cross-correlation coefficient is:



(a)

(b)

Figure 3.2: (a): Stochastic bifurcations for equation with case $\alpha < 0$. (b): Stochastic bifurcations for equation with case $\alpha > 0$

$$R_{XY} = \frac{\text{Cov}[XY]}{\sqrt{\text{Var}[X]\,\text{Var}[Y]}}.$$

The autocorrelation function (ACF) for the stochastic process X_t, $t \in (-\infty, +\infty)$, is:

$$r(t;\tau) = \frac{\text{Cov}[X_t X_{t+\tau}]}{\sqrt{\text{Var}[X_t]\,\text{Var}[X_{t+\tau}]}},$$

which for a stationary time-series reduces to:

$$r(\tau) = \frac{\text{Cov}[X_t X_{t+\tau}]}{\text{Var}[X_t]}.$$

3.5 Hurst analysis

Let us consider a stationary stochastic process $X_0, X_1, ...$ with stationary increments $X_t = Y_{t+1} - Y_t$. Y_t is a non-stationary process with $Y_0 = 0$. A $q'th$ order structure function $S_q(\tau)$ is the $q'th$ statistical moment:

$$S_q(\tau) \equiv E[Y_\tau^q].$$

If the motion Y_t is self-similar, the structure functions are power laws:

$$S_q(\tau) \propto \tau^{\zeta(q)},$$

where $\zeta(q) = Hq$, H is the self-similarity exponent for the fractional motion $Y(t)$ and the Hurst exponent for the increment fractional noise process X_t.

We calculate the Hurst exponent even if a process is neither Gaussian nor self-similar. The process can be investigated by Hurst R/S analysis [6]. The Hurst exponent (H) shows a quantitative measure of the persistence of time series. It was introduced by the British hydrologist Harold Edwin Hurst. He worked on the Nile River Dam Project in the 20th century and explored the water level and overflows.

The rescaled range (R/S) method is used to calculate the Hurst exponent. First step of this method is the division of time series with length N into n subseries $x = X_1, X_2, ..., X_n$ with length d. Then we calculate R/S statistics using the next algorithm for each subseries, $x = X_1, X_2, ..., X_n$:

1) find the mean E_m value, where $m = 1, ..., n$;

2) calculate standard deviation S_m:

$$S_m = \sqrt{\frac{1}{d}\sum_{i=1}^{d}(X_{i,m} - E_m)^2};$$

3) normalize the data ($X_{i,m}$) by subtracting the sample mean for $i = 1, d$:

$$Z_{i,m} = X_{i,m} - E_m;$$

4) create a cumulative time series for $i = 1, d$:

$$Y_{i,m} = \sum_{j=1}^{i} Z_{j,m};$$

5) calculate the range:

$$R_m = \max(Y_{1,m}, ..Y_{d,m}) - \min(Y_{1,m}, ..Y_{d,m});$$

6) rescale the range R_m/S_m;

7) finally, find the mean value of the rescaled range for subseries of length d:

$$(R/S)_d = \sqrt{\frac{1}{n}\sum_{m=1}^{n} R_m/S_m}.$$

After the analysis we plot a linear regression $\log(R/S)_d$ with respect to $\log(d)$. The slope coefficient of this regression is a value of the Hurst exponent. For the white noise process the Hurst exponent equal to $1/2$. If $H > 1/2$ the process is persistent, if $H < 1/2$ the process is anti-persistent.

Chapter 4

Early-warning signals for critical transitions

In this chapter we consider the early-warning signals for prediction of critical transitions in future. The prediction of tipping points in dynamical systems is extremely difficult problem. The system may have insignificant changes before the critical threshold is approached. But any system has similar signals and generic properties of behaviour near a tipping point despite differences in the details of each system. We can distinguish these certain symptoms of the tipping points and find early-warning indicators, which may predict the sudden shifts of the system from one state to another [7].

The stochastic dynamical systems are used to find the early-warning signals. We linearise systems around fixed point and add noise forcing.

The most important signal of the early warning is known as "critical slowing down". It is an indicator that the system loses resilience, thus, it may be tipped more easily into another state.

Figure 4.1 shows the model of critical slowing down. The ball represents the present state of the system and the potential well represents stable attractors. We approach the bifurcation point, the right potential well becomes smaller. At the bifurcation point it vanishes. The ball rolls into the left potential well and the recovery rate of the ball is slowing down. We also expect larger deviations as

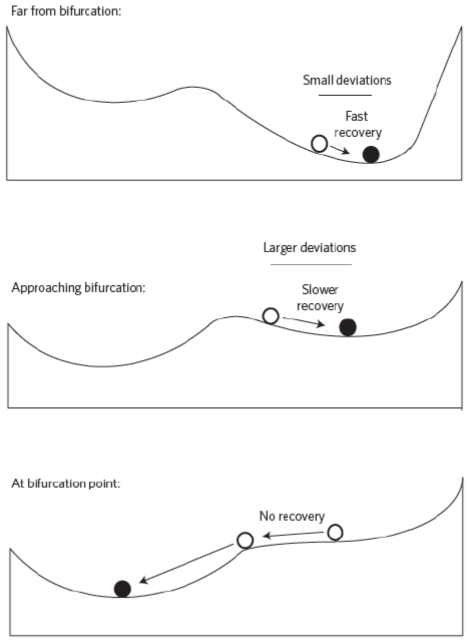


Figure 4.1: Model of critical slowing down.

bifurcation is approached. If we can detect the sign of slowing down, it would be possible to forecast the potential tipping point [8].

The system will take longer to recover from perturbations when it is close to a critical threshold [9]. We can use the recovery rate from small perturbations as an indicator of how close a system is to a bifurcation point. The critical slowing down leads to other possible early-warning signals: an increase in the variance of the fluctuations and an increase in the autocorrelation [10].

The memory of the system increases and the state at a present point is more similar to its previous state when it is near to a critical transition. We use the autocorrelation function to analyse the slowness of recovery. It defines that the data is more correlated with itself and the autocorrelation increases when the critical slowing down takes place [1].

The amplitude deviations may grow as the system approaches the critical transition and we can observe increasing skewness. We can use standard deviation to measure it:

$$\text{St.Dev.} = \frac{1}{n-1}\sum_{t=1}^{N}(z_t - \mu)^2,$$

where μ is the mean, z_t is the variable in time series in time t.

Chapter 5

Crises in financial markets

We consider description of power market in this chapter. Also we discuss the most known examples of financial crises.

5.1 Description of power market

The power (or electricity) is the main element that supports our modern life. We can not imagine our existence without it.

The electricity industry is relatively young. The earliest introduction of power market concepts and privatization to energy systems was in Chile in the early 1980s. The Chilean model had success in rationality of the power pricing, but had structural problems.

The prices of electricity depend on the balance between supply and demand. But other factors can influence the prices, for example the weather and the capacities of power plants, their current technical conditions. This balance between consumption and generation must exist constantly, 24 hours a day.

The power market differs from other commodity markets. The electricity cannot be stored, the prices of electricity are extremely unstable.

People originally used water power and coal for the production of electricity, but eventually sources were changed. Today nuclear power, natural gas, oil, coal, hydroelectric, wind generators, solar energy, geothermal steam and alternative resources such as cogeneration are mainly used for generation of power.

The economics varies considerably around the world, as a result the prices of electricity for the people of the various countries differ significantly. For all electricity markets demand is difficult to forecast and is almost completely insensitive to price fluctuations.

5.2 California power crisis and other financial crises

We can characterize power crises as sudden, lasting and large increases of prices. The most known example of power crisis was in California, which also known as the Western U.S. Energy Crisis. It took place in the beginning of the last decade (2000-2001 years). There was a shortage of electricity supply that was caused by market manipulations. It caused an 800% increase in wholesale prices on electricity spot markets.

It started in 1994 when California began to modify controls on its energy market and legislation was partially deregulate. The most important reason for deregulation was high prices for consumers. Electricity rates in California were on average 40 percent higher than the rest of the U.S in 1993 year. The restructuring plan was passed as California Assembly Bill in September 1996 year [11].

There were three large investor-owned utilities for serving the most of state's electricity users: Pacific Gas and Electric, Southern California Edison, San Diego Gas and Electric. The new changes included sale of a significant part of their electricity capacity and creation nonprofit corporations to control new markets, the California Independent System Operator (CAISO) and the California Power Exchange (PX). All sales and purchases of utilities had to go through the CAISO and the PX, all retail prices were controlled. It isolated economically a consumer from a producer of energy. Utilities were required to sell remains of their power generating capacity. This created dangerous situation for the investor-owned utilities [12].

A drought in the northwest states in the summer of 2001 year decreased the amount of hydroelectric power available in California. The weather conditions have caused an increase of demand and reduced the availability of electricity imports to California from neighbouring states. These states decreased their imports to California by an average 2,000 megawatts from 1999 to 2000. Demand for electricity rose faster than the generating capacity available to supply the markets. As a result, electricity supply was reduced.

There was increase of electricity prices in 2000. Prices for megawatt hour rose from 30 dollars in April to more than 100 dollars by June. By November, prices had increased to between 250 dollars and 450 dollars. The regulated utilities had to buy electricity in the wholesale markets and sell at much lower regulated prices in the retail markets.

As a result, the state's largest companies, Pacific Gas and Electric and Southern California Edison, declared bankruptcy in 2001.

Governor Davis declared a state of emergency in January 2001.

By June 2001, the seven-month California electricity crisis was over: wholesale prices fell to less than 50 dollars/MWh, demand dropped, new generating plants were coming on line.

A stock market crash is a sudden and often unexpected drop in stock prices with a significant loss of paper wealth. The most famous stock market crashes are the Black Monday, the Dot.com Bubble and 2008 Financial Crisis, which were explored by Cees Diks, Cars Hommes, Juanxi Wang (2012). The causes of them may be different, but they all share one common characteristic: all these events are in the state of financial bubble at first and jump to a long run depression state afterwards [8].

The Black Monday was the large stock market crash around the world that happened on 19 October 1987. The Dow Jones stock market index (Dow Jones Industrial Average) lost about 22% of its value. The crash of stock markets started in Hong Kong, spread western to Europe, then hit the United States. By the end of October, most of the major markets fell more than 20%. It was the largest one-day stock market decline in financial history.

The 2000 Dot.com Bubble (Information Technology Bubble) was a stock market bubble, which was caused by the rise of Internet sites and the tech industry in general. During the dotcom bubble of the late 1990s, the value of equity markets grew exponentially, with the technology-dominated NASDAQ index rising from under 1000 to 5000 between 1995 and 2000.

Chapter 6

Tipping points in financial markets

The main goal of our analysis is to review crisis in financial markets as a tipping point. For example, we explore California Power Crisis, the Black Monday, the "Dot.com bubble" and 2008 Financial Crisis. We need to analyze historical electricity prices to find early warning indicators. We also explore oil prices to find tipping points. We focus on sudden transitions in the market. One more objective is to research the possibility of detecting early warning signs. If it is possible, we will predict future sudden transitions in electricity markets that may cause a crisis. For our calculations and analysis we use a computational software program "Wolfram Mathematica 9.0". It is the most complete system for modern technical computing in the world.

6.1 Example of the stochastic dynamical system

Before starting our analysis of power crises we consider the next example of the stochastic dynamical system:

$$y' = r - y\left|1 - y\right| + \sigma w.$$

We can write this system: $y' = U'(y) + \sigma w$, where $U'(y) = r - y\left|1 - y\right|.$

Then find the function $U(y) = ry + \operatorname{sgn}[y-1](\frac{y^2}{2} - \frac{y^3}{3}) + \theta(1-y)/3.$

Consider the system:

$$y' = r - y\left|1 - y\right|.$$

To analyze this system we need to find the fixed points:

$$r - y\left|1 - y\right| = 0.$$

We have two cases for values of variable y.

First case: $1 - y > 0 \Rightarrow y < 1$.

$$r - y(1 - y) = 0,$$

$$y^2 - y + r = 0,$$

$$y_{1,2} = \frac{1 \pm \sqrt{1 - 4r}}{2}.$$

We take the value of $r < 0.25$, because $\sqrt{1 - 4r} > 0$. We get one stable and one unstable fixed points.

If $r = 0.25$ two fixed points unite into a half-stable fixed point at $y = 0.5$.

Second case: $1 - y < 0 \Rightarrow y > 1$.

$$r + y(1 - y) = 0,$$

$$y^2 - y - r = 0,$$

$$y_{1,2} = \frac{1 \pm \sqrt{1 + 4r}}{2},$$

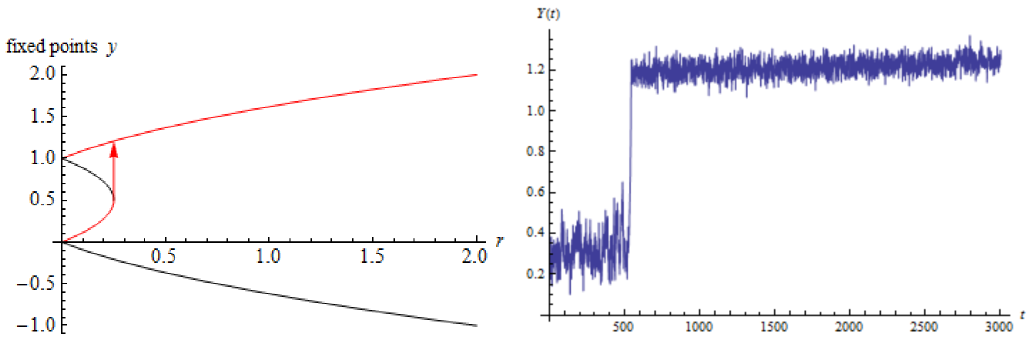
$$\sqrt{1 + 4r} > 1.$$

We have stable fixed point $\frac{1+\sqrt{1+4r}}{2}$.

The fixed points of our system are illustrated in Figure 6.1(a).

We explore stability of the fixed points. We get a stable fixed point $\frac{1-\sqrt{1-4r}}{2}$, an unstable fixed point $\frac{1+\sqrt{1-4r}}{2}$ and a stable fixed point $\frac{1+\sqrt{1+4r}}{2}$.

We plot a normal (Gaussian) distribution with mean μ=0 and standard deviation σ. We test the value of σ=0.15, which is responsible for size of noise. This



(a) (b)

Figure 6.1: (a): Fixed points of model $y' = r - x|1-x| + \sigma w$. Red curves show stable fixed points, black curves — unstable fixed points. (b): Normal (Gaussian) distribution with standard deviation σ=0.15.

is shown in Figure 6.1(b).

We consider $y' = F(y) + w$, where w is white noise, y_0 is a fixed point.

In Figure 6.1(b) we can see the jump approximately at the point $t = 530$.

$$F(y) = F(y_0) + F'(y_0)(y - y_0) + ...,$$

where $F(y_0) \to 0$, because y_0 is a fixed point, $F'(y_0)(y - y_0)$ is negative because y_0 is stable.

We get linearized equation:

$$y' = -a(y - y_0) + \text{noise}\,.$$

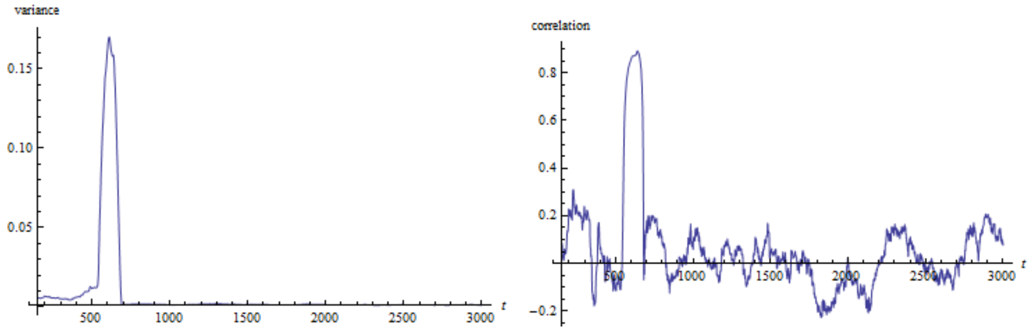
It is the Ornstein-Uhlenbeck (OU) process. The standard OU process are defined via stochastic differential equations of the form:

$$dY(t) = -a(y - y_0)dt + \sigma dB(t),$$

where $B(t)$ is a standard Brownian motion. The first term of equation is the damping term, and for $a > 0$ this causes anti-correlations since it prevents y to diffuse far from its value y_0 [13]. The fixed point becomes unstable $\Rightarrow a \to 0$.

$$\mathrm{Var}(Y(t)) = \frac{\sigma^2}{2a} \to +\infty.$$

We find variance and plot it in Figure 6.2(a). The rise of variance is a leading indicator of a tipping point. We can see in Figure 6.2(a) the significant increase approximately at the point $t = 530$. It displays an influence of the jump of our process at this point.



(a)

(b)

Figure 6.2: (a): Variance of $y' = r - y\,|1-y| + \sigma w$. (b): Correlation function of $y' = r - y\,|1-y| + \sigma w$.

The correlation is illustrated in Figure 6.2(b). We can also observe the jump approximately at the point $t = 530$.

The early warning of structural changes in any dynamical system driven through a bifurcation can only be obtained if increases in both variance and autocorrelation are observed before a tipping point [14] [15].

We should limit our variance and correlation functions before the time where the jump occurs. This is illustrated in Figures 6.3. We take the time period $t = [150, 530]$. There are increases in both variance and correlation functions at the point $t = 370$. We get early warning signals.

We can conclude that the tipping point in our example is approached approximately at $t = 530$.

In this example we take Gaussian normal distribution with white noise and the Hurst exponent $H = 0.5$. We explore the example of fractional Brownian motion

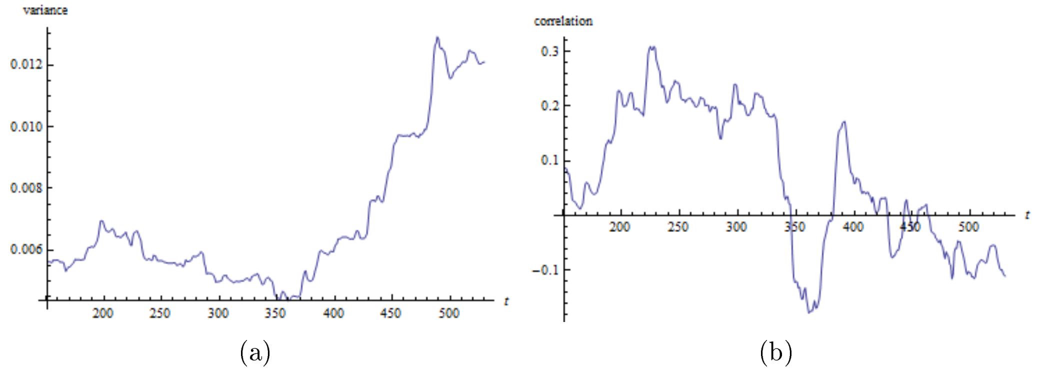


Figure 6.3: (a): Variance of $y' = r - y\,|1-y| + \sigma w$ before the tipping point. (b): Correlation function of $y' = r - y\,|1-y| + \sigma w$ before the tipping point.

process with the Hurst exponent that is not equal to 0.5. We find value of H in further analysis of California Power Crises. And in this example we also take, for example, $H = 0.368698$. A realization of this fractional Brownian motion is plotted in Figure 6.4. We take the same range. The standard deviation is $\sigma = 0.005$. In Figure 6.4 we can see the jump at the point approximately $t = 540$.

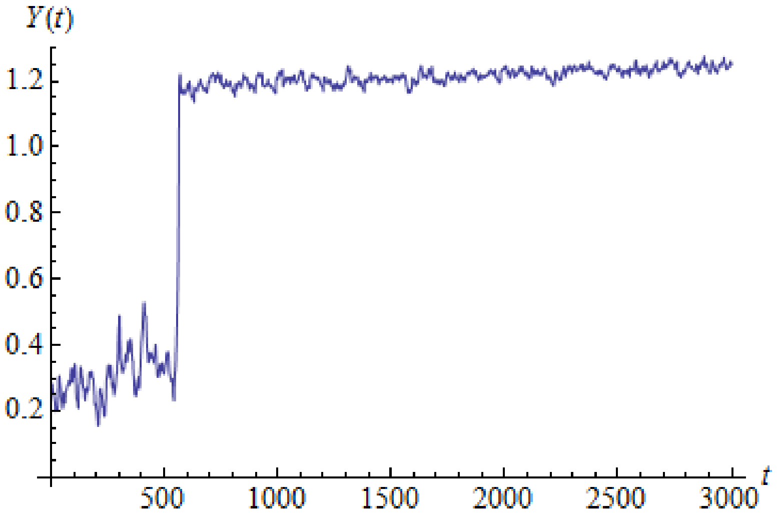
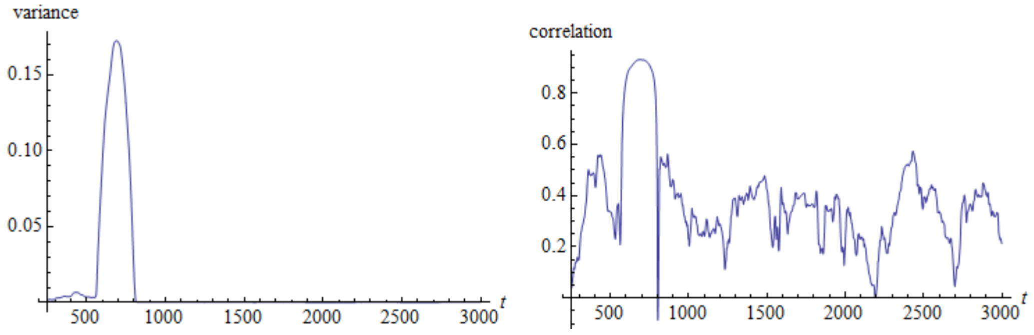


Figure 6.4: Fractional Brownian motion process with the Hurst exponent $H = 0.368698$.

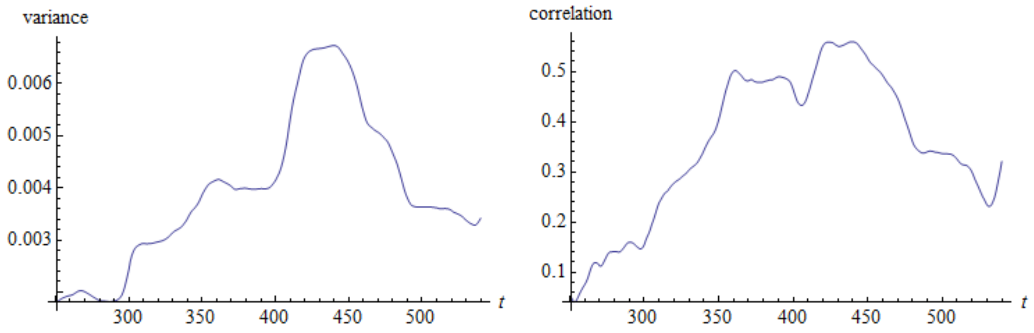
We plot variance and correlation functions to find early-warning signs. This can be seen in Figures 6.5.



(a) (b)

Figure 6.5: (a): Variance of Fractional Brownian motion process with the Hurst exponent $H = 0.368698$. (b): Correlation of Fractional Brownian motion process with the Hurst exponent $H = 0.368698$.

There is significant increase both in variance and correlation functions approximately at the point $t = 540$. It shows our jump of process at this point.



(a) (b)

Figure 6.6: (a): Variance of Fractional Brownian motion process with the Hurst exponent $H = 0.368698$ before the tipping point. (b): Correlation of Fractional Brownian motion process with the Hurst exponent $H = 0.368698$ before the tipping point.

We consider our variance and correlation functions before the time where the jump occurs. This is illustrated in Figures 6.6. We take the time period $t =$

$[250, 540]$ before the tipping point. There are increases in variance and correlation functions that start at the point $t = 300$. It is early warning signals.

In this example we use the fractional Brownian motion process and we get the tipping point at $t = 540$.

6.2 Estimation parameters systematically

Since we can not find the correct model from any physical and economic principles, then we can use an any model in our analysis. We take the next model:

$$x' = r - x\,|1 - x|\,.$$

Let us consider the example and estimate parameters for this model:

$$dx = F_r(x)dt + \sigma dB(t)$$

with

$$F_r(x) = \frac{1}{a}\left[x_c^2 r - x\,|2x_c - x|\right].$$

We find the fixed points of our model:

$$x^* = x_c - |x_c|\sqrt{1 - r}.$$

We take $r < 1$, because $\sqrt{1 - r} > 0$. We get a stable fixed point. Bifurcation is for $r = 1$ at $x = x_c$. We can make the conclusion that the tipping point appears when $x \to x_c$.

If we take data where the tipping point is, for example California Power Crises, we can choose x_c as the highest value of the signal prior to the tipping point.

The fixed point x^* changes with time, if the parameter r changes with time.

We use a running average to estimate $x^*(t)$:

$$x^*(t) \approx \langle x \rangle_{t,\Delta t} = \frac{1}{\Delta t}\int_{-\Delta/2}^{\Delta/2} x(s)ds,$$

and from this and the fixed point we get an estimate of the control parameter:

$$\hat{r}(t) = 1 - \left(\frac{\langle x \rangle_{t,\Delta t}}{x_c} - 1 \right)^2 .$$

We linearise $F_r(x)$ around the fixed point and get:

$$F_r(x) = -\theta(x - x^*) + \mathcal{O}(x - x^*)^2,$$

with

$$\theta = -F_r'(x^*) = -\frac{2\,|x_c|}{a}\sqrt{1-r}.$$

Then we write $x^* = x_c + \frac{a\theta}{2}$.

The linear equation $dx(t) = -\theta(x - x^*)dt + \sigma dB(t)$ is the Ornstein-Uhlenbeck (OU) process with auto-correlation function $e^{-\theta t}$.

$$\theta = -\log \rho.$$

It is the relation between the parameter θ and the one-step correlation ρ in the time series. We take the correlation ρ on windows of length Δt around t.

$$\hat{\theta}_{t,\Delta t} = -\log \hat{\rho}_{t,\Delta t}.$$

This gives the approximate equations:

$$\langle x \rangle_{t,\Delta t} = x_c + \frac{a\hat{\theta}_{t,\Delta t}}{2}.$$

The parameter σ can be chosen from formula:

$$\sigma = \sqrt{2\bar{\theta}}\,\mathrm{sdev}\left[x(t) - y(t)\right],$$

where $\mathrm{sdev}\left[x(t) - y(t)\right]$ is the standard deviation of the difference between real data $x(t)$ and curve $y(t)$, which is the solution of our equation without noise. $\bar{\theta}$ is the average of all the estimated $\hat{\theta}_{t,\Delta t}$ values.

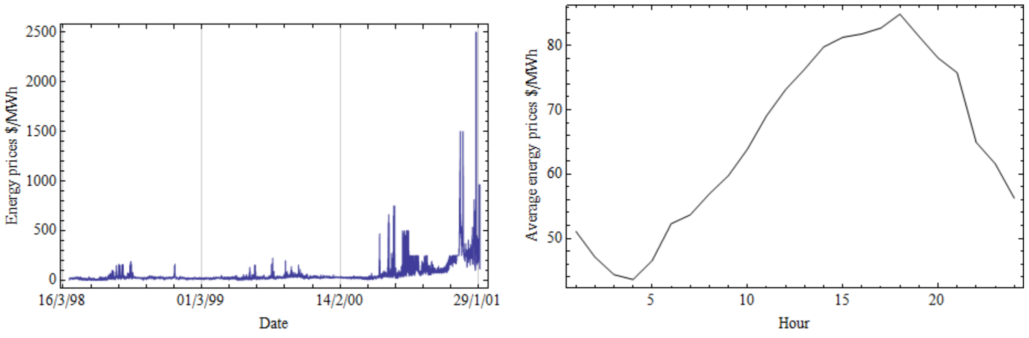
6.3 California power crisis

The California Electricity Crisis is well known example of power crisis in the world. How could this happen if deregulation was supposed to increase efficiency and reduce electricity prices? To answer the question we need to analyze power market and electricity prices in California.

We explore the time period from 1 April 1998, when California opened up a market for wholesale electricity.

We take prices for every hour since 1 April 1998, 0:00 until 31 January 2001, 24:00. This is illustrated in Figure 6.7(a). The data are received from The University of California Energy Institute (UCEI) [17]. We can see price spike in the end of January, 2001, when a state of emergency was declared in California.

We analyze an one-day period in Figure 6.7(b). We see from the figure that the spot price is higher in the evenings than in the mornings.

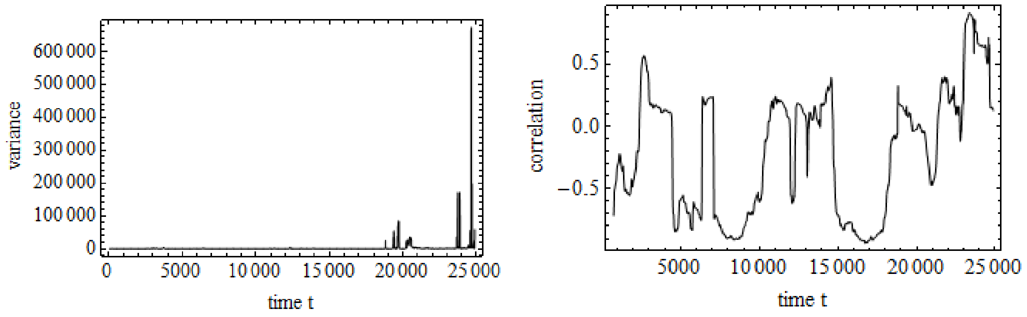


(a) (b)

Figure 6.7: (a): Energy prices in California per every hour since 1 April 1998, 0:00 until 31 January 2001, 24:00. (b): One-day period of prices in electricity market in California.

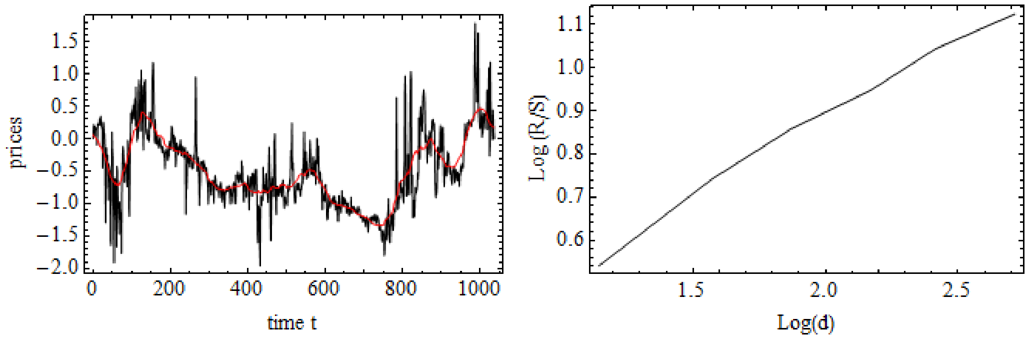
Variance and correlation functions are plotted in Figure 6.8(a) and Figure 6.8(b). There are jumps in variance and correlation functions at the point that corresponds to the end of January, 2001.

We model a system usually as a random walk-type process. The events in the system are independent and identically distributed (iid). But there are proofs



(a) (b)

Figure 6.8: (a): Variance for the values of energy prices. (b): Correlation function for the values of energy prices.



(a) (b)

Figure 6.9: (a): Average daily prices in California by taking the logarithm and subtracting a linear trend. The red curve is the moving average with $\Delta t = 1$. (b): Hurst R/S analysis of 1036 logarithmic returns of average daily prices.

that electricity price processes are not random walks [16]. It rather belongs to anti-persistent (mean-reverting) processes. We can prove it using means of the Hurst exponent (H) [6]. It shows a quantitative measure of the persistence of time series.

Our analyzing period of electricity prices includes 24 888 data points, 1037 days×24 hours.

We calculate average daily prices for this period and plot it by taking the logarithm and subtracting a linear trend. This can be seen in Figure 6.9(a). The red curve in Figure 6.9(a) illustrates the moving average with fixed time step $\Delta t = 1$ that removes the noise from price fluctuations.

We have the 1037 day-long sequence of average daily prices. We find divisors and divide the sample into subsamples of the equal length. We should take subseries of length $n \geq 10$ because rescaled range statistics for small value of n gives large estimation errors. Since the number 1037 has only two divisors greater than 10 we analyze a sample of length 1036, which has 7 divisors greater than 10: 14, 28, 37, 74, 148, 259, 518.

We selected the first 1036 and estimated using subseries of length n=14, 28, 37, 74, 148, 259, 518. After division we have n subseries of length d, where d — one of divisors.

We calculate all rescaled range statistics for every divisors using the algorithm of R/S statistics. We plot the $(R/S)_d$ against the subseries length d on a double-logarithmic paper in Figure 6.9(b).

We use a linear regression $\log(R/S)_d$ with respect to $\log(d)$. The slope coefficient of this regression is a value of the Hurst exponent. We get $H = 0.368698$. It means the process is anti-persistent [6].

6.4 Norwegian electricity spot market

Let us compare California's electricity prices and energy prices in Norway. We take time period of prices since 4 May 1992 until 27 August 2011. The data is received from the Data Administrator at Nord Pool Spot [18]. Prices are given hourly.

We analyze 20 years, it includes 7070 days×24 hours. All features of prices in Norway are shown in Figure 6.10(a). Prices have increased significantly since 1992 year. We can see the jump at the beginning of 2010 year.

We get 169680 data points that mean prices are given hourly. We consider an one-day period. We take average price for every hour during one day. This is illustrated in Figure 6.10(b). The highest price of day is approximately at 10 a.m. Also the price increases at 6 p.m.

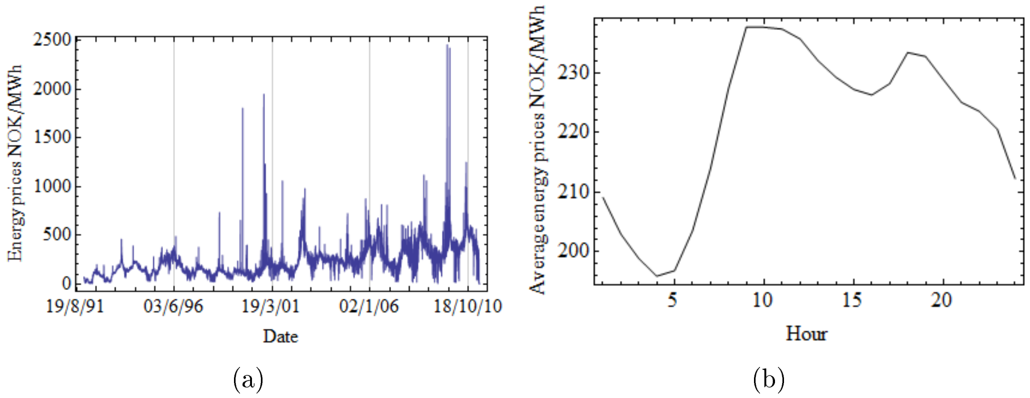


Figure 6.10: (a): Electricity prices in Norway for 1992-2011 years. (b): One-day period of prices in Norway.

In Figure 6.11(a) and Figure 6.11(b) we show variance and correlation functions for Norwegian electricity prices. There are 3 jumps in variance, but we do not have any increases in correlation function.

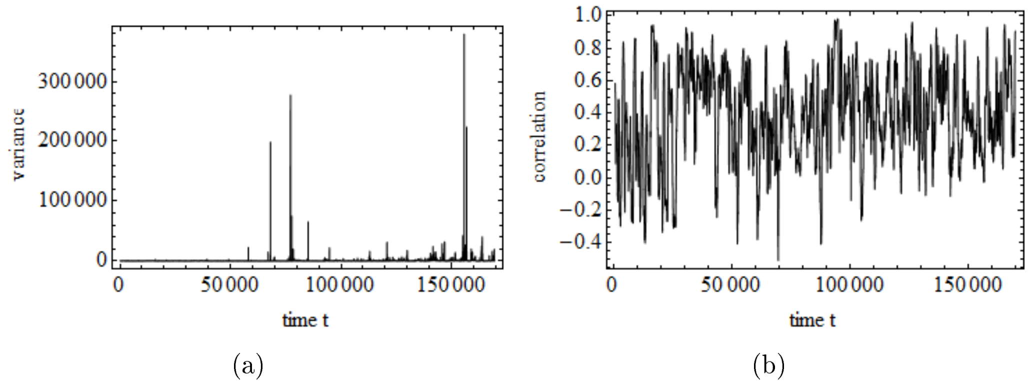
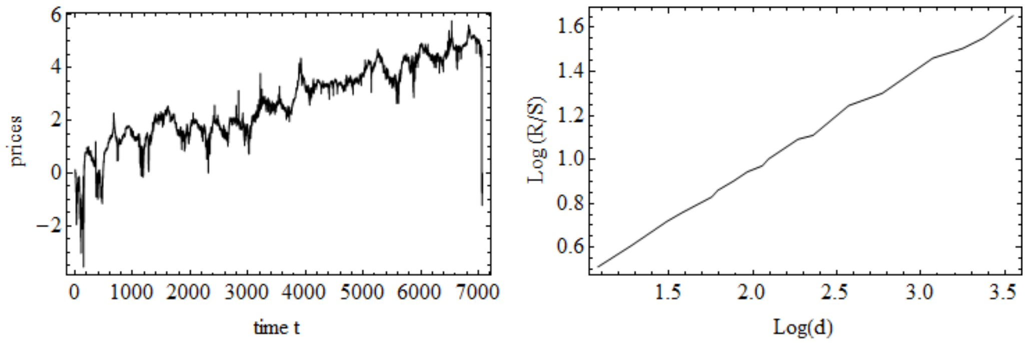


Figure 6.11: (a): Variance for the values of Norwegian electricity prices. (b): Correlation function for the values of Norwegian electricity prices.

We use the rescaled range (R/S) method to estimate the Hurst exponent.

We plot average daily prices in Norway for this period by taking the logarithm and subtracting a linear trend in Figure 6.12(a).

We have 7069 day-long sequence of average daily prices. We divide the sample



(a)

(b)

Figure 6.12: (a): Average daily prices in Norway by taking the logarithm and subtracting a linear trend. (b): Hurst R/S analysis of 7068 logarithmic returns of average daily prices.

into subsamples of equal length. Since the number 7069 does not have divisors we analyze a sample of length 7068, which has 18 divisors greater than 10 : 12, 19, 31, 38, 57, 62, 76, 93, 114, 124, 186, 228, 372, 589, 1178, 1767, 2356, 3534.

The algorithm of R/S statistics is analogous to that we took on the California market. We calculate all rescaled range statistics for every divisors and plot the $(R/S)_d$ against the subseries length d on a double-logarithmic paper. It is shown in Figure 6.12(b).

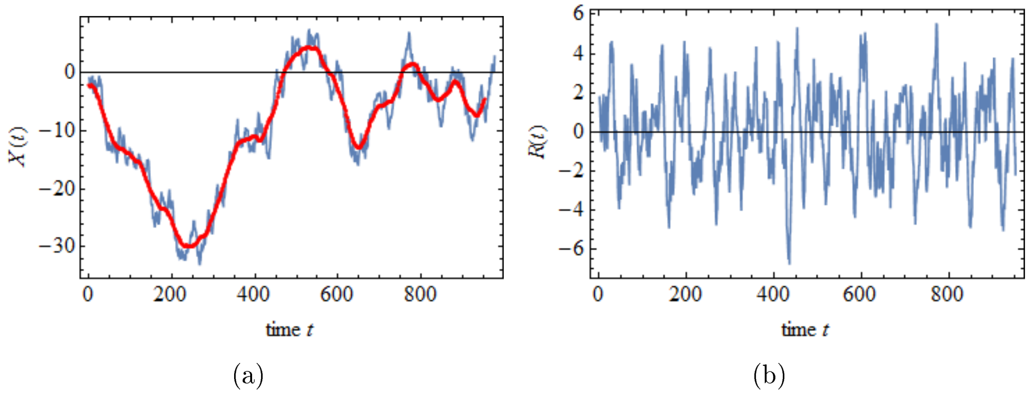
We get the Hurst exponent $H = 0.458081$. It means the process is anti-persistent.

There are not any tipping points and crises in Norwegian electricity prices.

6.5 Early warning signals for California power crisis

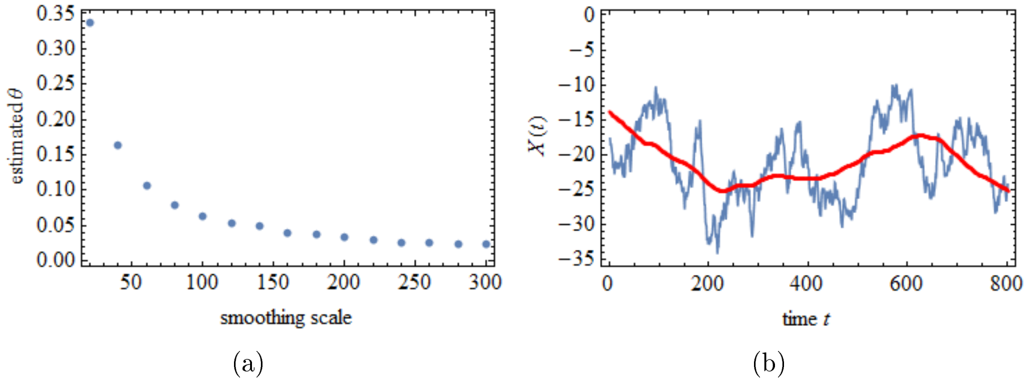
We simulate a random walk process $X(t)$ with normal Gaussian distribution, mean $\mu = 0$, standard deviation $\sigma = 1$ and length $n = 1000$. It is shown in Figure 6.13(a).

We estimate parameters of the Ornstein-Uhlenbeck process from our random



(a) (b)

Figure 6.13: (a): Random walk process with normal Gaussian distribution. The red curve is the moving average. (b): Residual of this random walk process.



(a) (b)

Figure 6.14: (a): Estimated θ on the smoothing scale. (b): Random walk process with the moving average (the red curve), where time step $\Delta t = 200$.

walk process $X(t)$, where μ means long-term mean, θ is a mean reversion speed, σ is a measure of the process volatility. If we estimate the θ in the OU process, then we get something close to zero, because $\theta = 0$ for a Brownian motion.

We get the next results: $\mu = -8.69676$, $\sigma = 10.2675$, $\theta = 0.00454413$.

Then define time step $\Delta t = 50$ and plot the moving average. It produces a list of averages over a window of length Δt. If we take a smaller value of Δt, then the curve of the moving average will be more similar to the curve of our process. It is illustrated in Figure 6.13(a), the red curve shows the moving average that makes a smoother time-series.

Financial data is more like a random walk process, but we need to take the Ornstein-Uhlenbeck process to find tipping points. So we take the residual of our process, it satisfies the OU process better.

If we make a smoothing of the signal $S(t)$ (the red curve), then the residual $R(t) = X(t) - S(t)$. It is well described by the Ornstein-Uhlenbeck process with θ corresponding to the time scale in the smoothing. The residual $R(t)$ is plotted in Figure 6.13(b).

Let us estimate parameters of the residual. We get $\mu = -0.0393581$, $\sigma = 2.27934$, $\theta = 0.103088$.

We look at this relationship. We take the statistical mean of the elements θ from the Ornstein-Uhlenbeck process on smoothing scale $t = [20, 300]$ using step $\Delta t = 20$. This can be seen in Figure 6.14(a).

Now we look at an example of random walk process with normal distribution, where the time step $\Delta t = 200$. It is plotted with the moving average in Figure 6.14(b).

We can write our data like *smooth* + *equation*, where *equation* is:

$$y' = a(r - y\,|1 - y|) + \text{noise}\,.$$

This equation has the fixed point $y^* = a\sqrt{1 - 4r}$.

The linear system

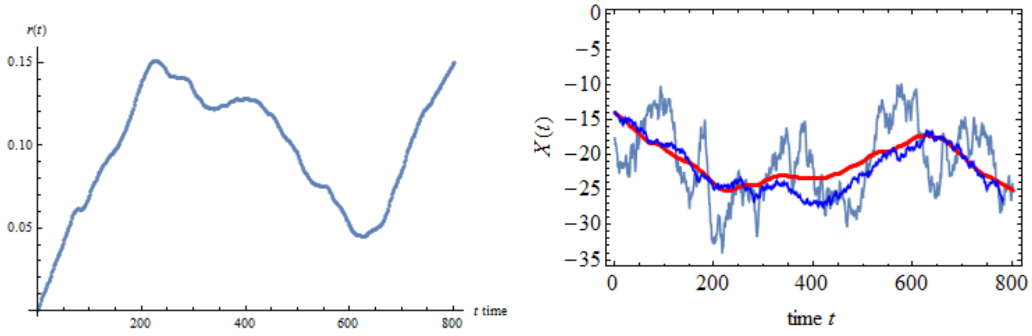
$$dx = a\sqrt{1 - 4r}dt + \sigma dB(t),$$

where $\theta = a\sqrt{1 - 4r}$.

We estimate the θ from the residual of the process, choose $r = 0.14$ because it is less than 0.25 and from estimation of θ we get value of $a = 0.0378933$.

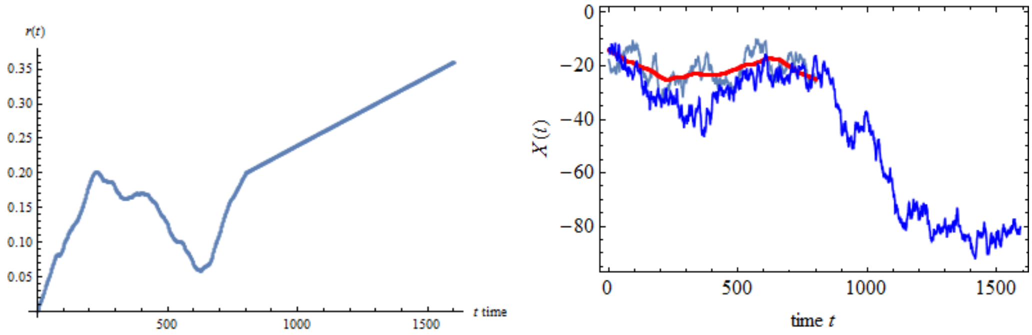
We construct function $r = r(t)$ from the moving average of our process. It is illustrated in Figure 6.15(a). Figure 6.15(b) shows model $y' = a(r - y\,|1 - y|) +$ noise with this $r = r(t)$. It is the dark blue curve. This figure also images our process (the light blue curve) and the moving average (the red curve).

Now we can construct a financial crash by going to the tipping point. We continue the curve $r = r(t)$ using the line $r(t) = 0.0002 \times t$, where $t = [800, 1600]$. It is illustrated in Figure 6.16(a) and we plot a new model using the new function $r = r(t)$ in Figure 6.16(b). It shows predictions of future behaviour for our example. We choose standard deviation $\sigma = 0.05$.



(a) (b)

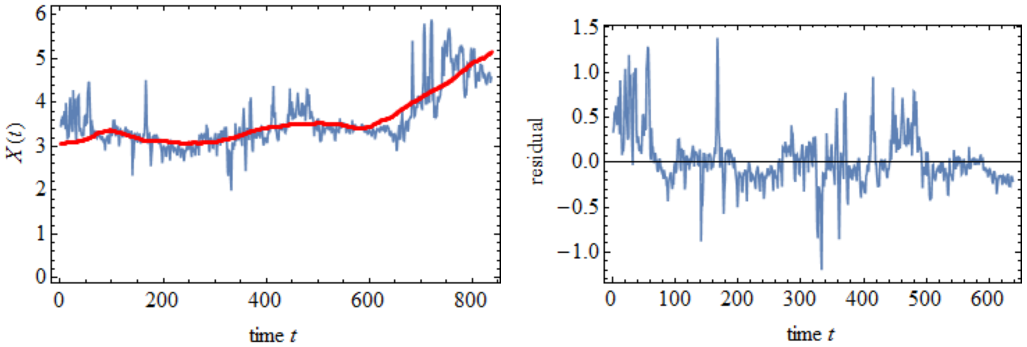
Figure 6.15: (a): Function $r = r(t)$ constructed from the moving average. (b): Model $y' = a(r - y\,|1 - y|) +$ noise constructed with the function $r = r(t)$ (the dark blue curve). Our random walk process and moving average are illustrated by the light blue curve and the red curve.



(a) (b)

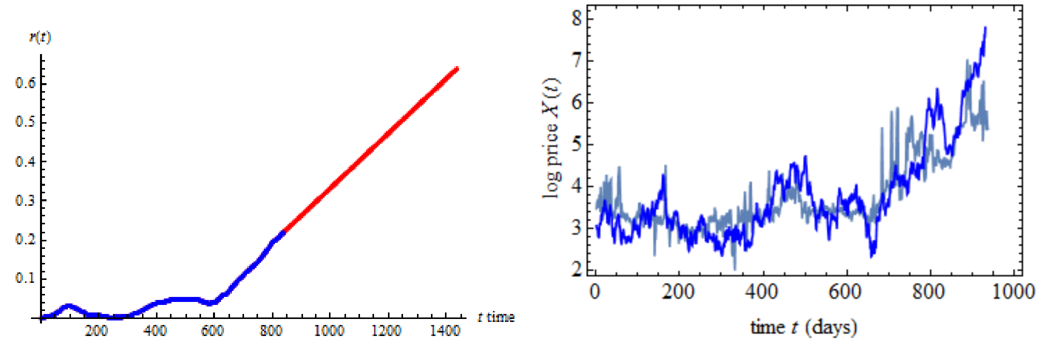
Figure 6.16: (a): New function $r = r(t)$ using the line $r(t) = 0.0002 \times t$ for $t = [800, 1600]$. (b): Model of our example constructed with the new function $r = r(t)$ with predictions of future (the dark blue curve).

Now we come back and look at our power data in California. The average daily prices of electricity in California are shown in Figure 6.17(a). We are interested in looking for an early warning signal before the crisis. We take only data around power crisis, calculating the logarithm. The sample size of our received data is equal to 800 elements. We also plot the moving average with the time step $\Delta t = 200$. As in example, we make a smoothing of the signal and get the residual before the crisis in Figure 6.17(b).



(a) (b)

Figure 6.17: (a): Average daily prices of electricity in California. The red curve is the moving average with time step $\Delta t = 200$. (b): Residual of the average daily prices of electricity in California.



(a) (b)

Figure 6.18: (a): Function $r = r(t)$ constructed from the moving average. (b): New model with predictions with the function $r = r(t)$ constructed from the moving average (the dark blue curve). The initial real data of average daily prices is shown by the light blue curve.

We estimate the θ from the residual of the Ornstein-Uhlenbeck process and from equation $\theta = a\sqrt{1-4r}$ get $a = 0.0378933$.

We plot the function $r = r(t)$ of the moving average. It is illustrated in Figure 6.18(a). We continue this curve using the line $r(t) = 0.0007 \times t$, where $t = [800, 1400]$. The red line shows it.

Now we can construct a financial crash by going to the tipping point. The plot of a new model with the function $r = r(t)$ constructed from the moving average is shown in Figure 6.18(b) by dark blue curve. We choose standard deviation $\sigma = 0.03$. Figure 6.18(b) also shows initial real data of average daily prices. We stop the curves before the tipping point.

Figure 6.19 shows the logarithm of data of average daily prices with the statistical mean of the elements. The shaded area has the lower bound of value of mean with 25%, and the higher bound of value of mean with 75%. We estimate the value r. Since $r < 0.25$ this is before the tipping point. We get the value $\theta = 0.0251356$.

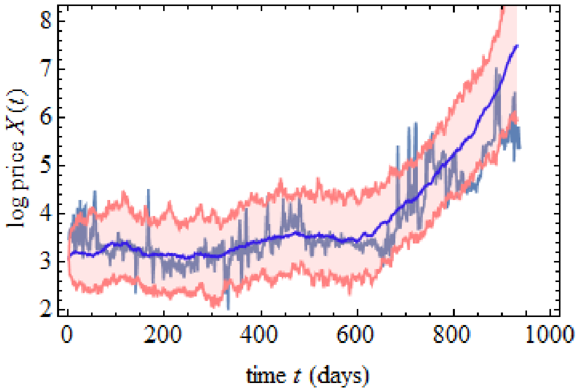
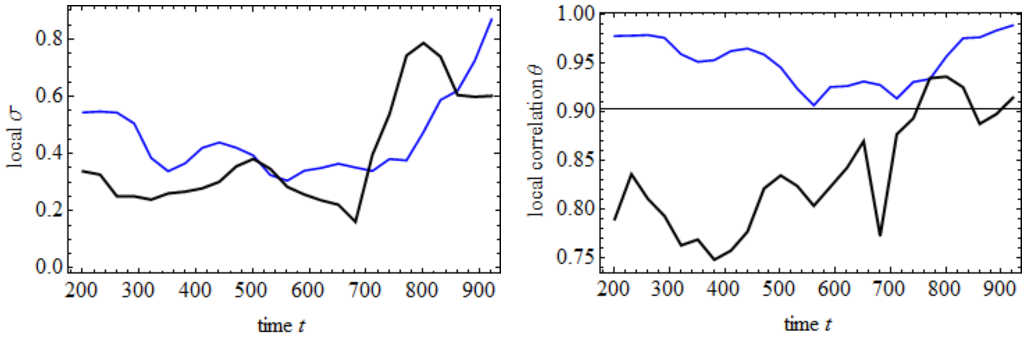


Figure 6.19: Logarithm of data of average daily prices with statistical mean of elements of our constructed model.

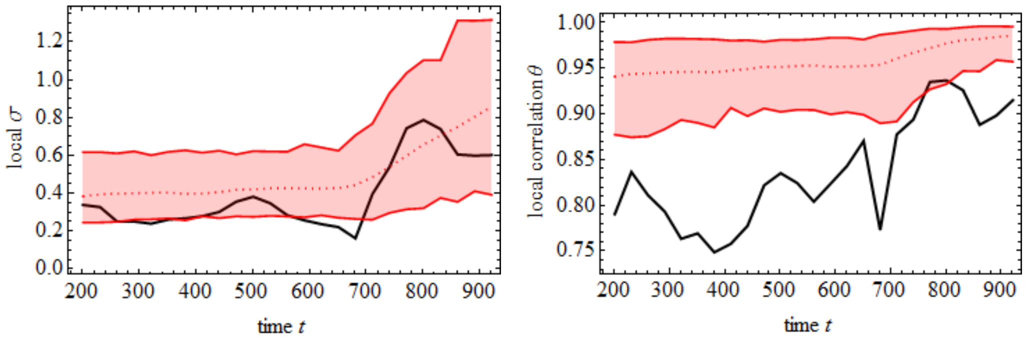
We plot the local standard deviation σ and the local correlation scale θ in Figure 6.20(a) and 6.20(b). The black curves are illustrated σ and θ for real data, blue curves — for new constructed model. We can observe significant increases of variance and correlations function for real data at the point $t \approx 700$.

Figures 6.21(a) and 6.21(b) show Monte Carlo method for the local standard



(a) (b)

Figure 6.20: (a): Local standard deviation. (b): Local correlation scale. The black curves are illustrated σ and θ for real data, blue curves — for our constructed model.

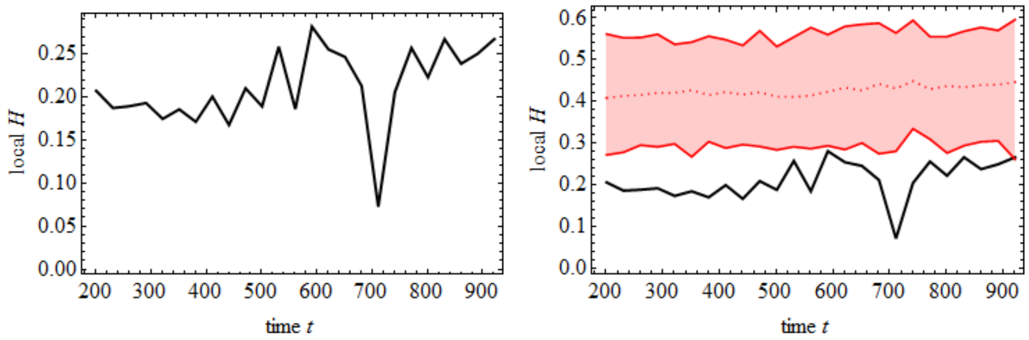


(a) (b)

Figure 6.21: (a): Monte Carlo simulations for the local standard deviation σ. (b): Monte Carlo simulations for the local correlation scale θ. The black curve is illustrated real data, the dashed curve is the statistical mean of σ and θ for constructed model.

deviation σ and the local correlation scale θ for real data. Monte Carlo method is a technique that involves using random variables to approximate the probability of certain outcomes. It builds the models of possible results by substituting a range of values (a probability distribution) and lets us see all the possible variants of future.

We plot the mean of σ and θ. The shaded area has the lower bound of value of



(a) (b)

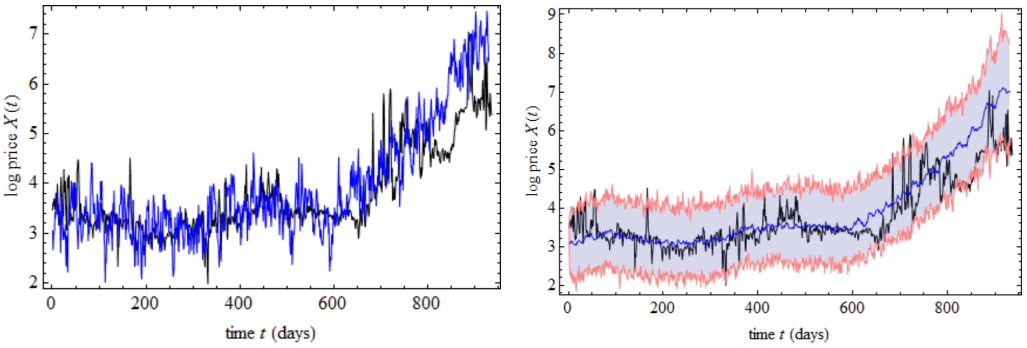
Figure 6.22: (a): Local Hurst exponent. (b): Monte Carlo simulations for the local Hurst exponent. The black curves are illustrated real data, the dashed curves are the statistical mean of H for constructed model.

mean with 25%, and the higher bound of value of mean with 75%. The curve of local standard deviation σ is almost completely included within the shaded area. There is an increase of possible values of σ for our constructed model. It is the early warning signal. The curve of local correlation θ does not coincide with shaded area. These possible values of the local correlation are plotted for the white noise process with the Hurst exponent $H = 1/2$. We have an anti-persistent process with $H < 1/2$.

We calculate the local Hurst exponent, using a function for de-trended fluctuation analysis. It is shown in Figure 6.22(a). Figure 6.22(b) illustrates Monte Carlo method for the local Hurst exponent. The curve of the local Hurst exponent does not coincide with shaded area.

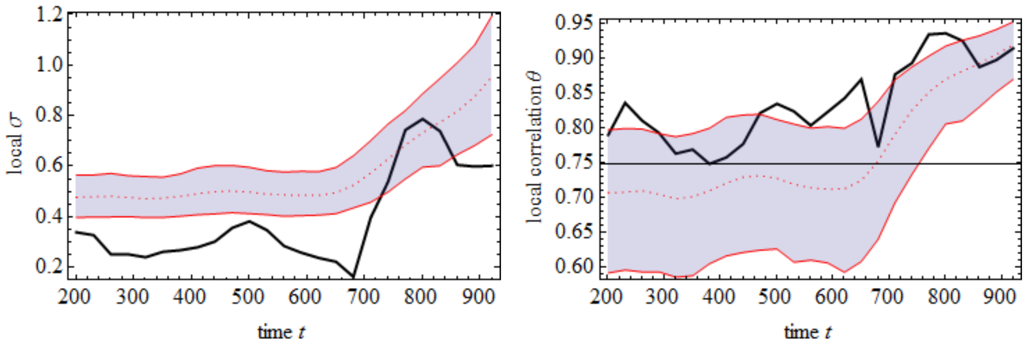
Now we change the white noise process with the value of Hurst exponent $H = 1/2$ to Fractional Brownian Motion Process with the Hurst exponent $H = 0.45$. Let us construct a financial crash by going to the tipping point. We plot the new model with the function $r = r(t)$ constructed from the moving average in Figure 6.23(a). This figure also shows real data of average daily prices. We choose standard deviation $\sigma = 0.07$. We stop the curves before the tipping point.

Figure 6.23(b) shows the logarithm of data of average daily prices with the statistical mean of the elements. The shaded area has the lower bound of value of



(a) (b)

Figure 6.23: (a): New model with the function $r = r(t)$ constructed from the moving average (the blue curve). The real data of average daily prices (the black curve). (b): Logarithm of data of average daily prices with the statistical mean of the elements of our constructed model.



(a) (b)

Figure 6.24: (a): Monte Carlo simulations for the local standard deviation σ. (b): Monte Carlo simulations for the local correlation scale θ. The black curve is illustrated real data, the dashed curve is the statistical mean of σ and θ for constructed model.

mean with 25%, and the higher bound of value of mean with 75%. We plot Monte Carlo method for the local standard deviation σ and the local correlation scale θ for real data in Figure 6.24(a) and 6.24(b). There are significant increases both in local standard deviation σ and the local correlation function θ for real data and for possible values of our constructed model at the point $t \approx 700$. It is the early warning signal for California electricity prices.

We calculate the local Hurst exponent using Monte Carlo method. This is illustrated in Figure 6.25. There is increase in the local Hurst exponent function. It is also at the point $t \approx 700$. The real data of the local Hurst exponent coincides with the shaded area.

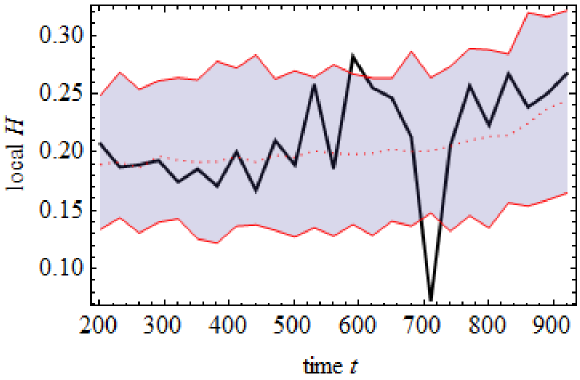


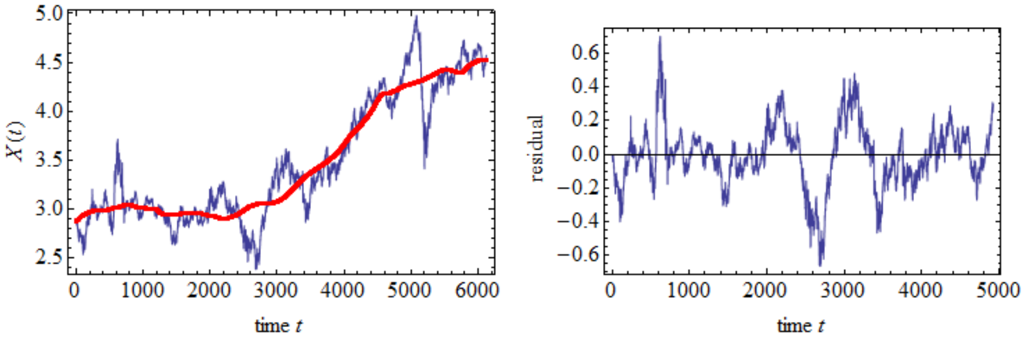
Figure 6.25: Monte Carlo simulations for the local Hurst exponent of fractional Brownian motion process.The black curve is illustrated real data, the dashed curve is the statistical mean of H for constructed model.

6.6 Oil prices

The crude oil is the world's most actively traded commodity.

Let us analyze oil prices of 1986-2014 years that are given per day. The time period, which we take, is 7323 days. There are something like 260 trading days in a year. The analyzed data are received from US. Energy Information Administration [19]. The average oil prices per day by taking the logarithm are illustrated in Figure 6.26(a). There is big oil crisis, a sharp increase approximately at the point $t = 4800$ that corresponds July of 2008. We also plot the moving average with the time step $\Delta t = 1200$. The residual of oil prices is received by subtracting the moving average from our data. This is illustrated in Figure 6.26(b).

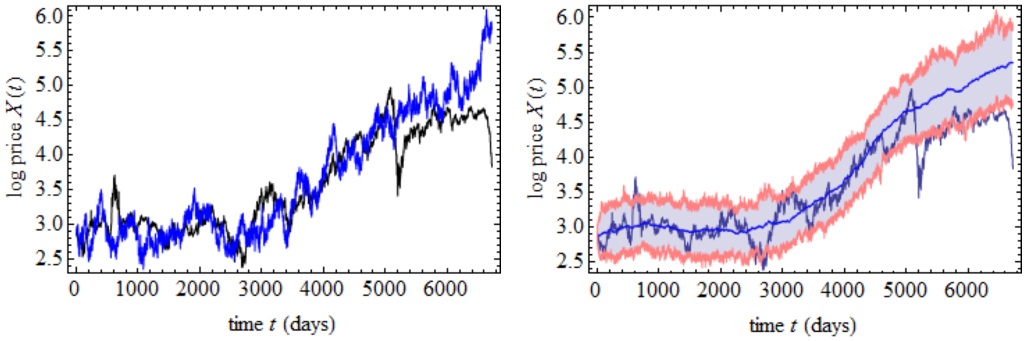
We construct a crisis by going to the tipping point. Figure 6.27(a) shows our new model of oil prices with the function $r = r(t)$ constructed from the moving



(a) (b)

Figure 6.26: (a): Average daily oil prices with the moving average (the red curve) by taking the logarithm for 1986-2014 years. (b): Residual of average oil prices per day before crisis.

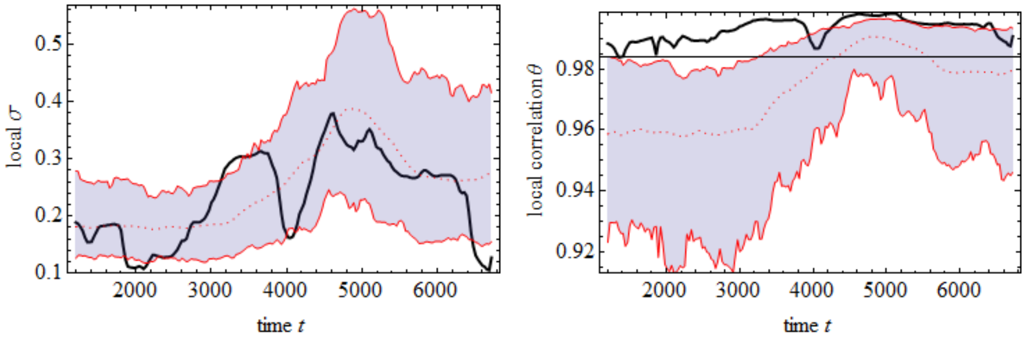
average. We use Fractional Brownian Motion Process with the Hurst exponent $H = 0.3$. The standard deviation $\sigma = 0.02$. Next Figure 6.27(b) illustrates the logarithm of data of average prices with the statistical mean of the elements of our model. The shaded area has the lower bound of value of mean with 25% and the higher bound of value of mean with 75%.



(a) (b)

Figure 6.27: (a): The blue curve is the new model of oil prices constructed with the function $r = r(t)$. The black curve is the initial real data of daily oil prices. (b): Logarithm of average daily oil prices with statistical mean of our model.

We plot the local standard deviation σ and the local correlation scale θ for real



(a)

(b)

Figure 6.28: (a): Monte Carlo simulations for the local standard deviation σ. (b): Monte Carlo simulations for the local correlation scale θ.

data in Figure 6.28(a) and 6.28(b). It is the black curve. We used Monte Carlo method. We plot the mean of σ and θ for our constructed model. It is the dashed curves. And the shaded area has the lower bound of value of mean with 25%, and the higher bound of mean with 75%. We get increases of local standard deviation and local correlation functions for real data. The curves have jumps at the point $t \approx 4000$. We compare it with our constructed model. There are also jumps in both functions σ and θ at the point $t \approx 3200$. We get the early-warning signal for crisis.

6.7 Stock market crashes. Standard&Poor 500 index

We consider three well-known stock market crashes, the Black Monday 1987, the 2000 Dot.com Bubble, 2008 Financial crisis.

We use the Standard&Poor 500 (S&P 500) index to find early-warning signals prior to these crises. This is shown in Figure 6.29. The S&P 500 index is famous stock market index that includes 500 American companies with large market capitalizations. It represents the U.S. stock market. In Figure 6.29 the index rises at points that correspond to three crises.

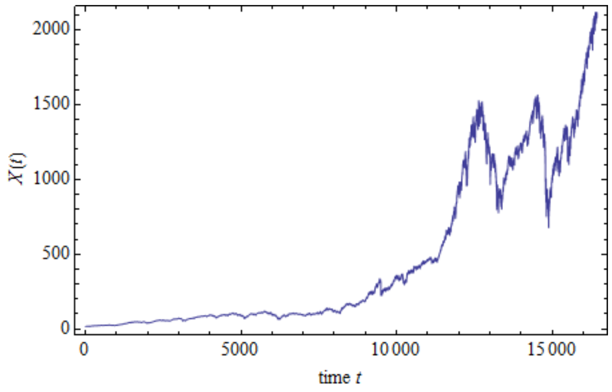


Figure 6.29: Standard&Poor 500 index for 1950-2015 years.

Figure 6.30(a) shows the daily Standard&Poor 500 index by taking the logarithm and subtracting linear trend. There are increases approximately at the points $t = 7000$, $t = 9500$. It corresponds to the Black Monday and the Dot.com Bubble. We use only the data before 2008 Financial Crisis to find the early-warning signals before the crash. The red curve in Figure 6.30(a) is the moving average with $\Delta t = 4000$.

We subtract the moving average from our financial data and get the residual of time series before crisis. This curve is plotted in Figure 6.30(b).

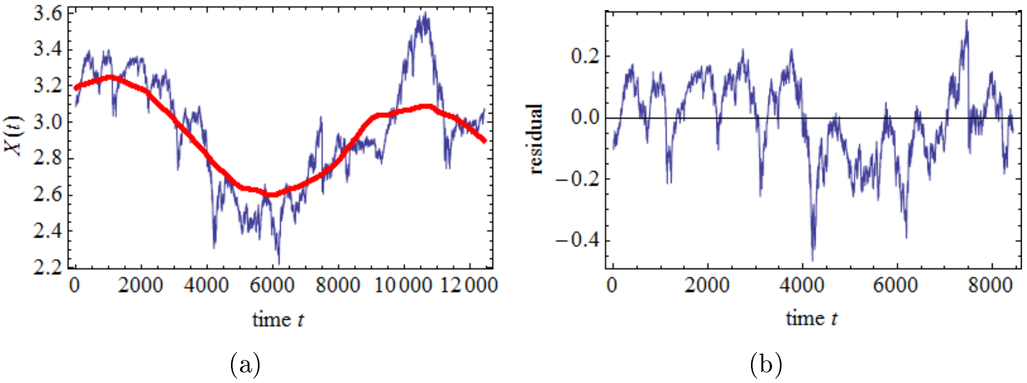
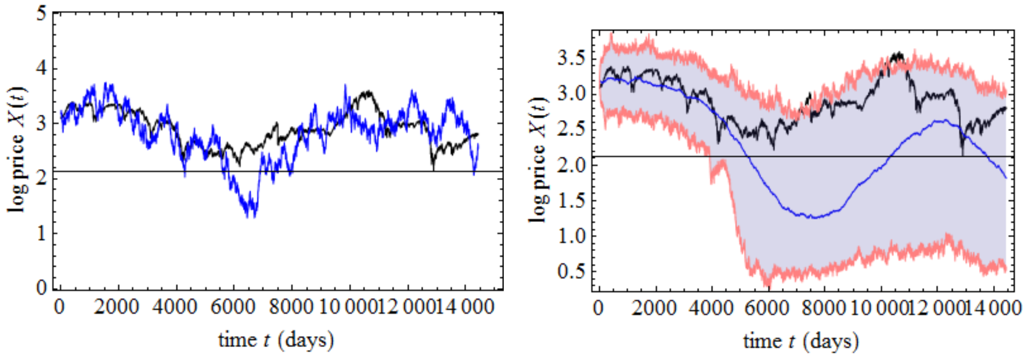
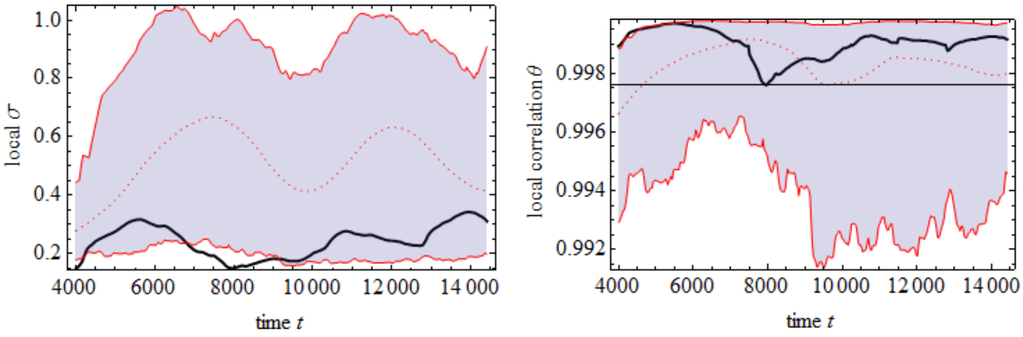


Figure 6.30: (a): Daily Standard&Poor 500 index by taking the logarithm and subtracting linear trend. The red curve is the moving average with $\Delta t = 4000$. (b): Residual of time series before crisis.



(a) (b)

Figure 6.31: (a): The blue curve is the new model of crisis constructed with function $r = r(t)$ from the moving average. The black curve is the initial real data of S&P 500 index. (b): Logarithm of financial data with statistical mean of elements of our constructed model.



(a) (b)

Figure 6.32: (a): Monte Carlo simulations for the local standard deviation σ. (b): Monte Carlo simulations for the local correlation scale θ. Black curves are for real data. The mean of σ and θ for our constructed model is dashed curves. And the violet area has the lower bound of value of mean with 25%, and the higher bound of mean with 75%.

Figure 6.31(a) shows the new model of crisis constructed with the function $r = r(t)$ from the moving average. This is illustrated by the blue curve. We use a model of white noise process with standard deviation $\sigma = 0.025$. The black curve shows the initial real data of S&P 500 index.

We plot the logarithm of financial data with statistical mean of elements. This can be seen in Figure 6.31(b). We use the Monte-Carlo method with the lower bound of value of mean with 25% and the higher bound of value of mean with 75%.

Monte Carlo simulations for the local standard deviation σ and for the local correlation scale θ are illustrated in Figures 6.32. There are insignificant increases both in standard deviation function and the correlation function at the point $t = 13000$ before the tipping point. It is the early warning signal of the imminent crash.

Chapter 7

Conclusion

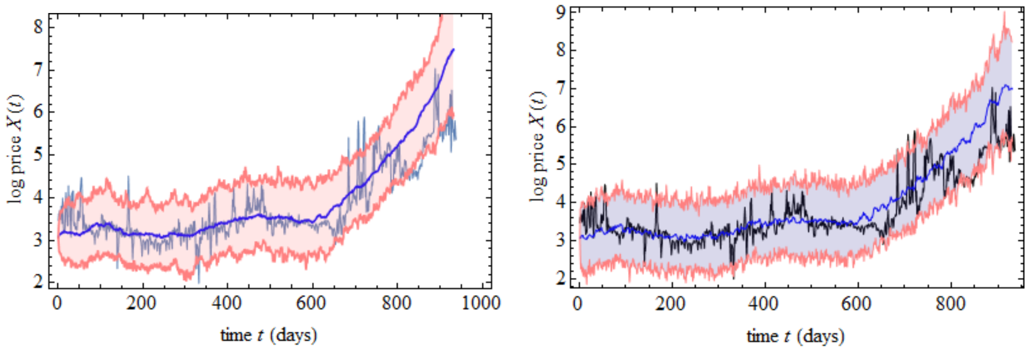
The main point of this thesis was to review crises in financial markets within the framework of a tipping point. We have defined that the tipping point is a point in a system where a sudden transition between two states occurs, and we have described power crises and other financial crises as bifurcations in stochastic dynamical systems.

We have applied stochastic models for electricity spot prices, oil prices and stock market prices. We also estimated parameters for these models.

Our results show that the local variance and the local autocorrelation may grow as the system approaches the critical transition, the so-called early-warning signals. Our analysis confirm that these indicators can be seen before critical threshold is reached. We use the standard deviation and the correlation scale as our main indicators.

We explore Norwegian electricity prices for the years 1992-2011 and we can conclude that there are no evidence of any tipping points nor power crises in these data. The Hurst R/S analysis shows that this process is anti-persistent, i.e. $H < 0.5$.

We compare California energy prices and find early-warning signals. Our analysis also reveals that this process is anti-persistent. We model a random walk process with normal Gaussian distributions, consistent with a fractional Brownian motion process with the Hurst exponent $H = 0.45$. We compare two models. This is illustrated in Figures 7.1, and we got almost identical results. We can use both models for our analysis.



(a)

(b)

Figure 7.1: (a): Model of random walk process with normal Gaussian distribution. (b): Model of fractional Brownian motion with $H = 0.45$.

We found the significant increases of the standard deviation and the correlation function before California energy crisis in both models. It shows critical slowing down prior to the tipping point.

The model for oil prices of 1986-2014 years also has early-warning signals. Here we used fractional Brownian motion process with the Hurst exponent $H = 0.3$. There are increases in the standard deviation and the correlation function for our constructed model and for real data. It corresponds to crisis in 2008.

We also analysed the Standard&Poor 500 index, which displays stock market prices. We modelled log returns as a white noise process with the Hurst exponent $H = 0.5$. Monte-Carlo method for the standard deviation and the correlation function shows early-warning signals for financial crisis of 2008 year.

We can remark these models are suitable for forecasting spot prices. This analysis can be performed on other financial data. It would be interesting to continue research in this direction and predict crises in future.

Chapter 8

Bibliography

[1] Timothy M. Lenton, *Early warning of climate tipping points.* Nature Climate change, 201-209, 2011.

[2] H.A. Dijkstra, *Nonlinear climate dynamics.* Cambridge University Press, 2013.

[3] Steven H. Strogatz, *Nonlinear dynamics and chaos - with applications to physics, biology, chemistry and engineering 1st edn.* Perseus Books Publishing, 1994.

[4] Kristoffer Rypdal, *Lectures for the course MAT-3213 Climate Dynamics.* Department of mathematics and statistics, UiT, 2014.

[5] Ross A. Maller, Gernot Muller, Alex Szimayer, *Ornstein-Uhlenbeck processes and extensions.* Handbook of financial time series, Springer: p.421-437, 2009.

[6] R. Weron, B. Przybylowicz, *Hurst analysis of electricity price dynamics.* Physica A 283, 2000.

[7] M. Scheffer, J. Bascompte, W. A. Brock, V. Brovkin, S. R. Carpenter, V. Dakos, H. Held, E. H. van Nes, M. Rietkerk, G. Sugihara, *Early-warning signals for critical transitions.* Nature, 461: 53-59, 2009.

[8] Cees Diks, Cars Hommes, Juanxi Wang, *Early Warning Signals for Critical Transitions in Finance.* CeNDEF Working Paper Universiteit van Amsterdam, 2012.

[9] V. Dakos, Sonia Kefi, M. Reitkerk, E. H. van Nes, M. Scheffer, *Slowing down in spatially patterned ecosystems at the brink of collapse.* The University of Chicago Press, 2011.

[10] S.R. Carpenter, W.A. Brock, *Rising variance: a leading indicator of ecological transition.* Ecol. lett. 9, 311-318, 2006.

[11] The Congress of the United States, Congressional Budget Office, *Causes and lessons of the California electricity crisis.* CBO papper, 2001.

[12] James L. Sweeney, *The California electricity crisis: lessons for the future.* Stanford: Hoover Institution., 2002.

[13] Martin Rypdal, Ola Løvsletten, *Modelling electricity spot prices using mean-reverting multifractal processes.* Physica A 392, 2013.

[14] Peter D. Ditlevsen and Sigfus J. Johnsen, *Tipping points: Early warning and wishful thinking.* Geophysical research letters, vol.37, L19703, 2010.

[15] T. M. Lenton, V. N. Livina, V. Dakos, E. H. van Nes, M. Scheffer, *Early warning of climate tipping points from critical slowing down: comparing methods to improve robustness.* Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 370, number 1962, 1185-1204, 2012.

[16] I. Simonsen, *Volatility of power markets.*Physica A 355, 10-20, 2005.

[17] *http://www.ucei.berkeley.edu/* University of California Energy Institute.

[18] *http://www.nordpoolspot.com/* Nord Pool Spot.

[19] *http://www.eia.gov/* U.S. Energy Information Administration.

Appendices

Appendix A

Source code of example of tipping point

This appendix includes the source code of example of stochastic dynamical system. It is used to find the tipping points of this system. We take a random walk process with the Hurst exponent $H = 0.5$, then fractional Brownian motion process with $H = 0.368698$.

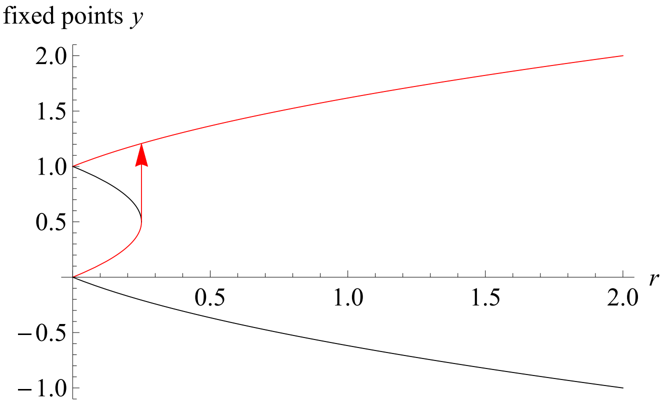
We consider the dynamical system $y' = r - y \mid 1 - y \mid + \sigma\, w$. This can be written as $y' = U'(y) + \sigma\, w$, where $U(y) = r\, y\ + \mathrm{sgn}(y) \left(y^2 / 2 - y^3 / 3\right) + \theta(1 - y)/3$.

```
rand = RandomReal[NormalDistribution[0, 0.1], 5 * 31];
rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
ifun = Interpolation[rand];
```

```
σ = 0.15;
```

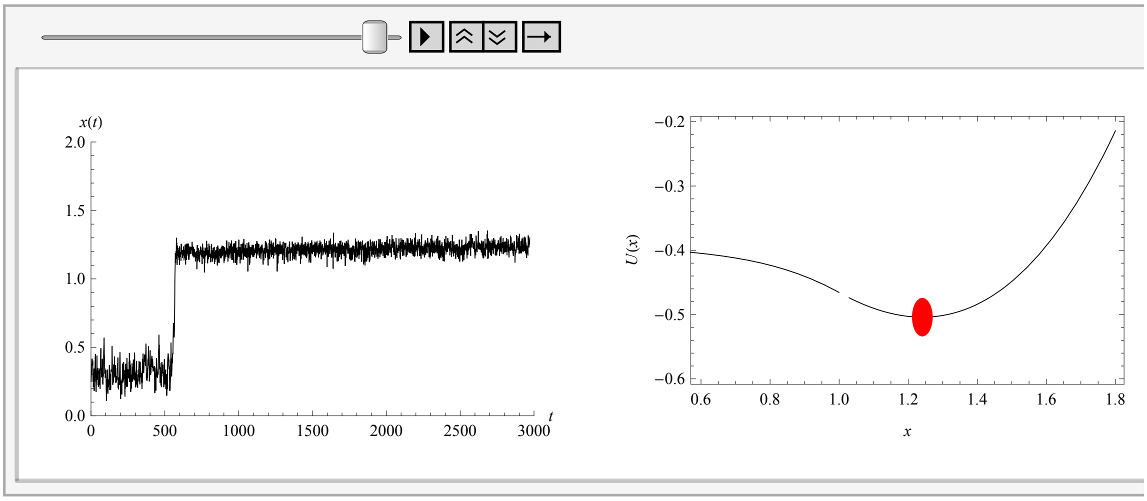
```
i[r_] := r y + Sign[y - 1] (y^2 / 2 - y^3 / 3) + UnitStep[1 - y] / 3;
f1[r_] := 1/2 (1 - √(1 - 4 r));
f2[r_] := 1/2 (1 + √(1 - 4 r));
f3[r_] := 1/2 (1 - √(1 + 4 r));
f4[r_] := 1/2 (1 + √(1 + 4 r));
```

```
PL1 = Plot[f1[r], {r, 0, 2}, PlotRange → All, PlotStyle → Red];
PL2 = Plot[f2[r], {r, 0, 2}, PlotRange → All, PlotStyle → Black];
PL3 = Plot[f3[r], {r, 0, 2}, PlotRange → All, PlotStyle → Black];
PL4 = Plot[f4[r], {r, 0, 2}, PlotRange → All, PlotStyle → Red];
line = Graphics[{ Red, Arrow[{{0.25, 0.5}, {0.25, f4[0.25]}}]}];
Show[{PL1, PL2, PL3, PL4, line},
 AxesLabel → {"r", "fixed points  y"}, AxesStyle → Directive[14]]
```

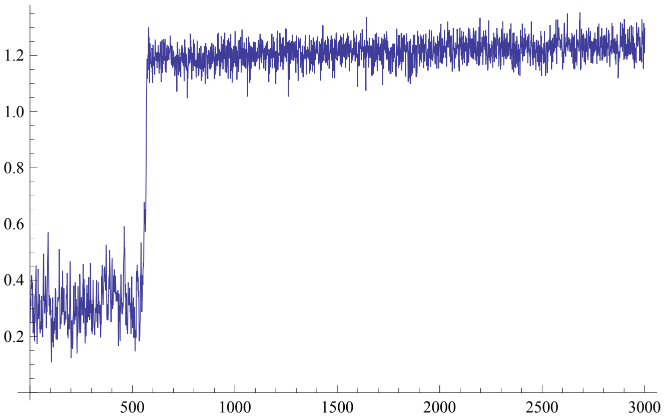


```
r = 0.2;
tab = {0.25};
rliste = {r};
pliste = {};
Monitor[
 Do[
   rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
   rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
   ifun  = Interpolation[rand];
   (* xxxxxxxx *)
   r = r + 0.001;
   rliste = Append[rliste, r];
   y0 = Last[tab];
   s = NDSolve[
     {y′[x] == r - Sqrt[(1 - y[x]) ^2] * y[x] + ifun[x + 1], y[0] == y0}, y, {x, 0, 30}];
   mid = Flatten[Evaluate[y[x] /. s] /. x → Range[30]];
   tab = Join[tab, mid];
   p1 = ListPlot[tab, PlotRange → {{0, 3000}, {0, 2}},
     Background → None, AxesStyle → Black, PlotStyle → Black,
     Joined → True, AxesLabel → {"t", "x(t)"}];
   PL1 = Plot[-i[r], {y, 0, 1.8}, Background → None, Axes → False,
     Frame → True, FrameStyle → Black, PlotStyle → Black];
   L = {PL1};
   point1 = {f4[r], -i[r] /. y → f4[r]};
   point2 = {f2[r], -i[r] /. y → f2[r]};
   point3 = {f1[r], -i[r] /. y → f1[r]};
   If[Element[point1, Reals],
    L = Append[L, Graphics[{Red, Disk[point1, 0.03]}]];
   ];
   If[Element[point2, Reals],
    L = Append[L, Graphics[{Blue, Disk[point2, 0.03]}]];
   ];
   If[Element[point3, Reals],
    L = Append[L, Graphics[{Green, Disk[point3, 0.03]}]];
   ];
   p2 = Show[L, PlotRange → {-0.6, -0.2}, FrameLabel → {"x", "U(x)"}];

   p = GraphicsGrid[{{p1, p2}}, ImageSize → 600, Background → None];
   pliste = Append[pliste, p];
   , {t, 1, 100}];
 , t]
```

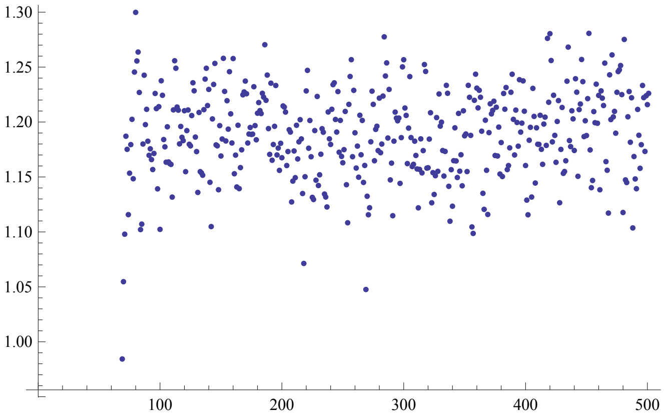
```
ListAnimate[pliste]
```



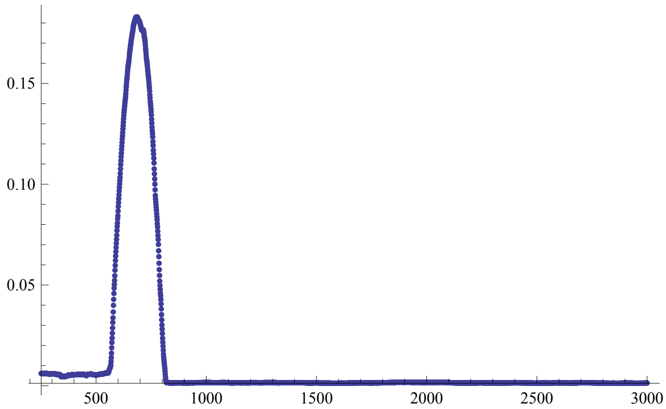
```
ListPlot[tab, Joined → True]
```



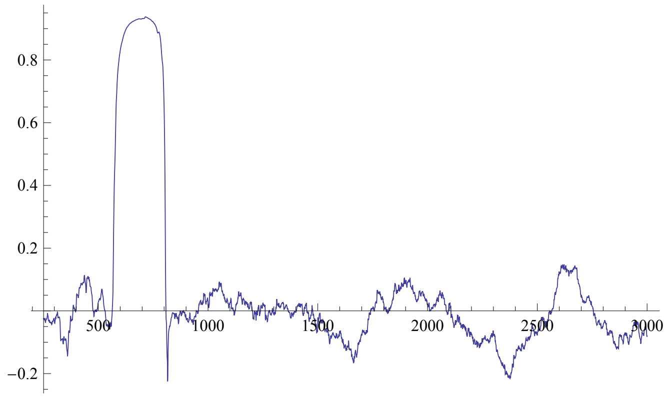
```
ListPlot[tab[[500 ;; 1000]]]
```



```
Δt = 250;
V[t_] := Variance[tab[[t - Δt ;; t]]]
variancelist = Table[{t, V[t]}, {t, 251, 3000}]
varPlot1 = ListPlot[variancelist, PlotRange → All]
```



```
n = 12;
corlist =
  Table[{t, Correlation[Drop[tab[[t - Δt ;; t]], n], Drop[tab[[t - Δt ;; t]], -n]]},
   {t, 251, 3000}];

ListPlot[corlist, Joined -> True, PlotRange → All]
```



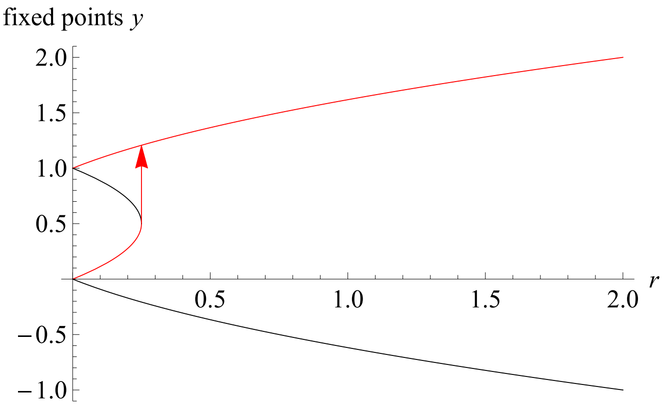
We consider the dynamical system $y' = r - y\,|\,1 - y\,| + \sigma\, w$. This can be written as $y' = U'(y) + \sigma\, w$, where $U(y) = r\, y\; + \mathrm{sgn}(y)\left(y^2/2 - y^3/3\right) + \theta(1 - y)/3$.

```
data = RandomFunction[FractionalBrownianMotionProcess[.368698], {0, 5 * 31, 1}];
data = Drop[data["Path"][[All, 2]], 1];
data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
ifun  = Interpolation[data];
```

```
σ = 0.15;
```

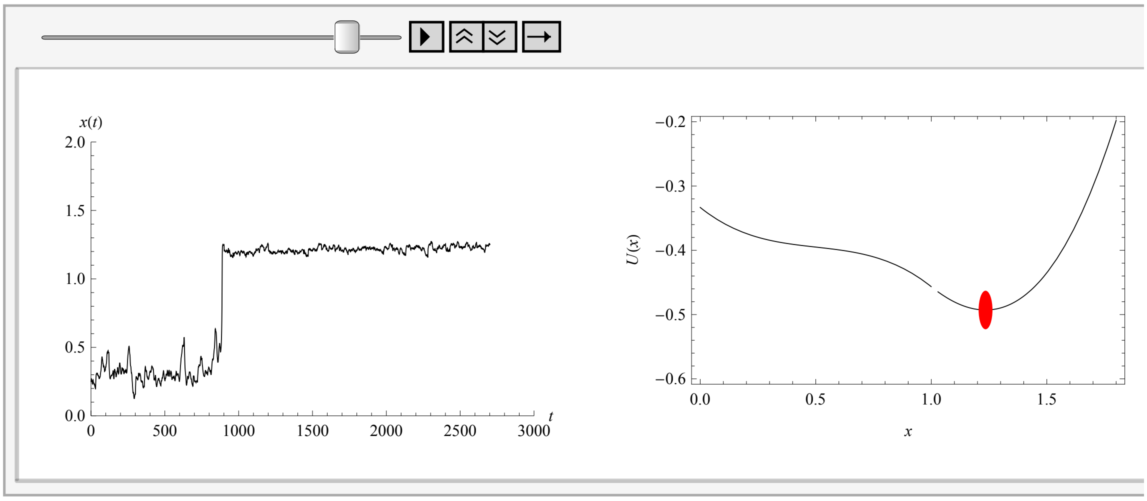
```
i[r_] := r y + Sign[y - 1] (y^2 / 2 - y^3 / 3) + UnitStep[1 - y] / 3;
          1
f1[r_] := — (1 - √1 - 4 r );
          2
          1
f2[r_] := — (1 + √1 - 4 r );
          2
          1
f3[r_] := — (1 - √1 + 4 r );
          2
          1
f4[r_] := — (1 + √1 + 4 r );
          2
```

```
PL1 = Plot[f1[r], {r, 0, 2}, PlotRange → All, PlotStyle → Red];
PL2 = Plot[f2[r], {r, 0, 2}, PlotRange → All, PlotStyle → Black];
PL3 = Plot[f3[r], {r, 0, 2}, PlotRange → All, PlotStyle → Black];
PL4 = Plot[f4[r], {r, 0, 2}, PlotRange → All, PlotStyle → Red];
line = Graphics[{ Red, Arrow[{{0.25, 0.5}, {0.25, f4[0.25]}}]}];
Show[{PL1, PL2, PL3, PL4, line},
 AxesLabel → {"r", "fixed points  y"}, AxesStyle → Directive[14]]
```

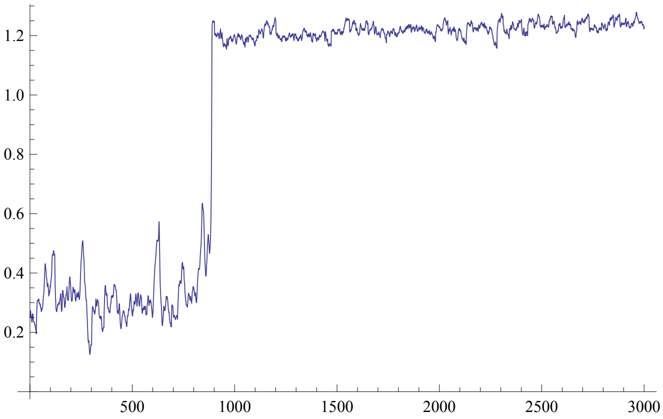


```
r = 0.2;
σ = 0.005;
tab = {0.25};
rliste = {r};
pliste = {};
Monitor[
 Do[
   data =
    RandomFunction[FractionalBrownianMotionProcess[.368698], {0, 5 * 31, 1}];
   data = Drop[data["Path"][[All, 2]], 1];
   data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
   ifun = Interpolation[data];
   (* xxxxxxxx *)
   r = r + 0.001;
   rliste = Append[rliste, r];
   y0 = Last[tab];
   s = NDSolve[{y′[x] == r - Sqrt[(1 - y[x]) ^2] * y[x] + σ * ifun[x + 1], y[0] == y0},
     y, {x, 0, 30}];
   mid = Flatten[Evaluate[y[x] /. s] /. x → Range[30]];
   tab = Join[tab, mid];
   p1 = ListPlot[tab, PlotRange → {{0, 3000}, {0, 2}},
     Background → None, AxesStyle → Black, PlotStyle → Black,
     Joined → True, AxesLabel → {"t", "x(t)"}];
   PL1 = Plot[-i[r], {y, 0, 1.8}, Background → None, Axes → False,
     Frame → True, FrameStyle → Black, PlotStyle → Black];
   L = {PL1};
   point1 = {f4[r], -i[r] /. y → f4[r]};
   point2 = {f2[r], -i[r] /. y → f2[r]};
   point3 = {f1[r], -i[r] /. y → f1[r]};
   If[Element[point1, Reals],
    L = Append[L, Graphics[{Red, Disk[point1, 0.03]}]];
   ];
   If[Element[point2, Reals],
    L = Append[L, Graphics[{Blue, Disk[point2, 0.03]}]];
   ];
   If[Element[point3, Reals],
    L = Append[L, Graphics[{Green, Disk[point3, 0.03]}]];
   ];
   p2 = Show[L, PlotRange → {-0.6, -0.2}, FrameLabel → {"x", "U(x)"}];

   p = GraphicsGrid[{{p1, p2}}, ImageSize → 600, Background → None];
   pliste = Append[pliste, p];
   , {t, 1, 100}];
 , t]
```

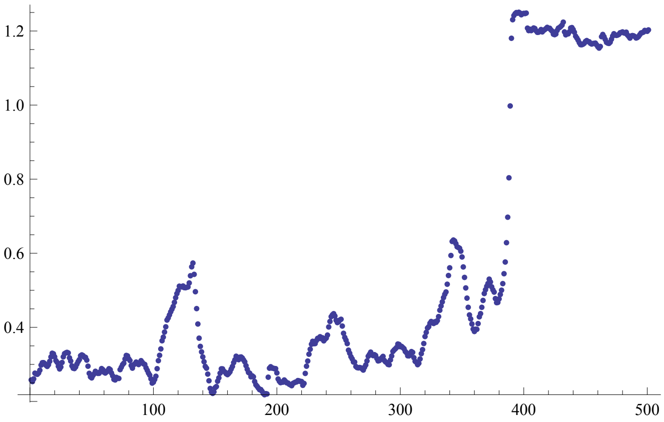
```
ListAnimate[pliste]
```

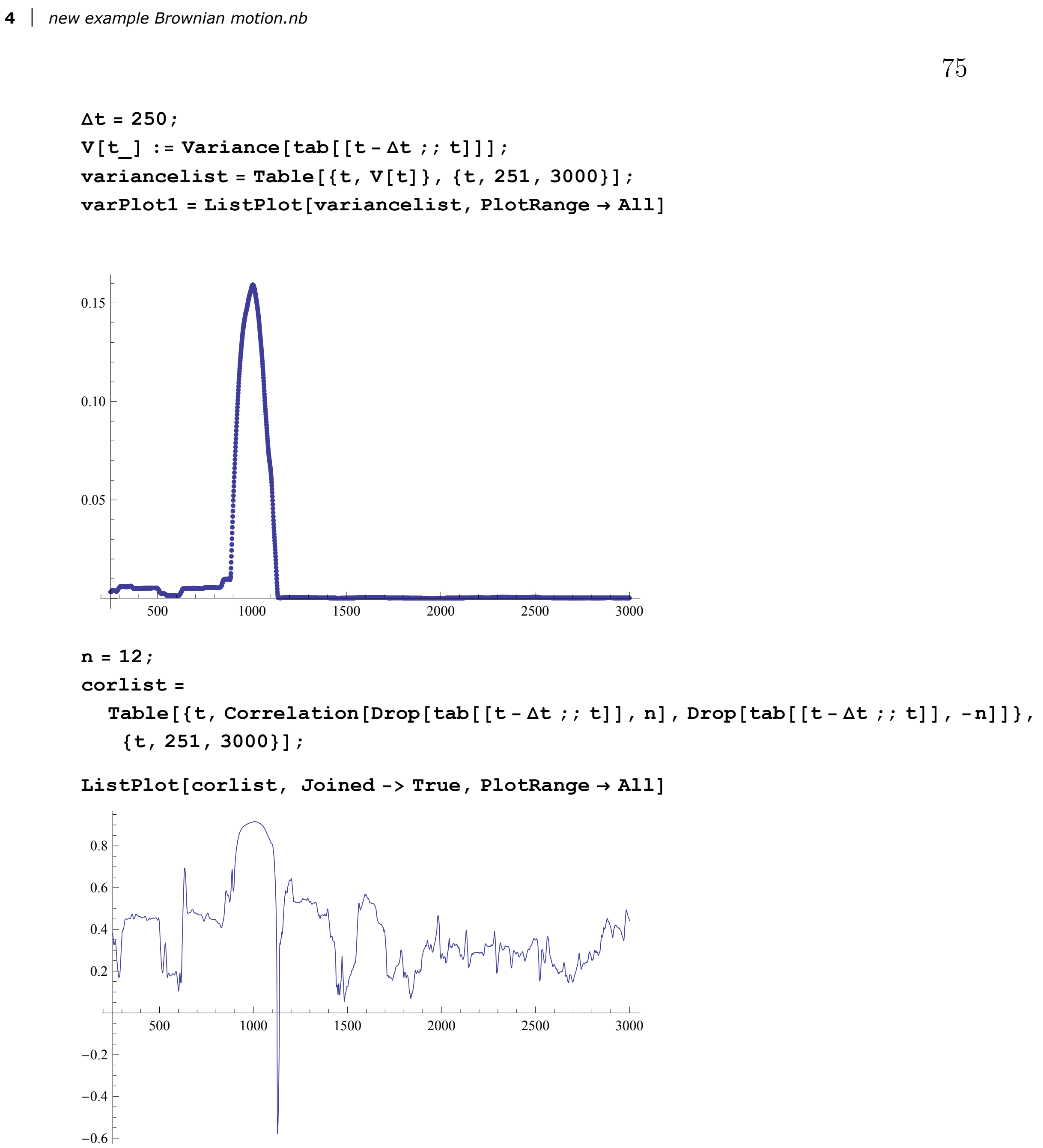


```
ListPlot[tab, Joined → True]
```



```
ListPlot[tab[[500 ;; 1000]]]
```





4 | *new example Brownian motion.nb*

```
Δt = 250;
V[t_] := Variance[tab[[t - Δt ;; t]]];
variancelist = Table[{t, V[t]}, {t, 251, 3000}];
varPlot1 = ListPlot[variancelist, PlotRange → All]
```

```
n = 12;
corlist =
  Table[{t, Correlation[Drop[tab[[t - Δt ;; t]], n], Drop[tab[[t - Δt ;; t]], -n]]},
   {t, 251, 3000}];
```

```
ListPlot[corlist, Joined -> True, PlotRange → All]
```

Appendix B

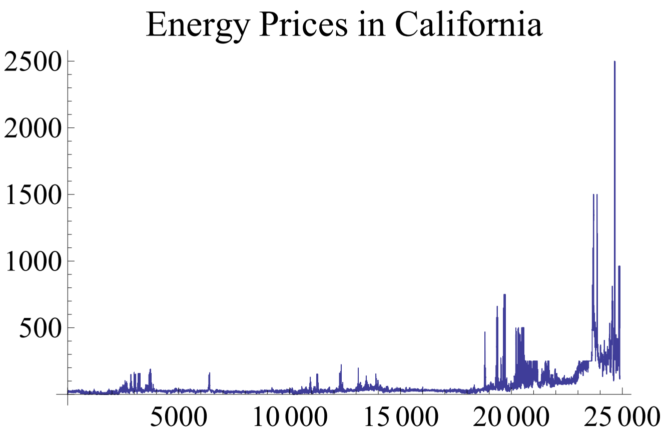
Source code of California power crisis

This part includes analysis of California electricity prices. The code plots average prices per day and per hour in California for 1998-2001, variance and correlation function. Also we use the algorithm of R/S statistics and find the Hurst exponent.

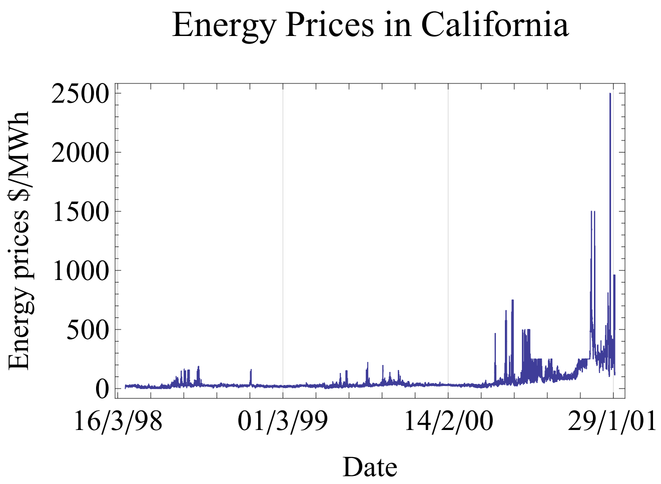
```
x = ToExpression[StringSplit[
    ReadList["C:/Users/i/Desktop/California electricity crisis/calpxnew.txt",
     String]][[All, 3]]]
```

```
pricehourly = Flatten[x];
```

```
ListPlot[pricehourly, Joined → True, PlotRange → All, LabelStyle → {16},
 PlotLabel → Style["Energy Prices in California", 20, Black]]
```



```
DateListPlot[pricehourly, {1998, 4, 1, 1}, Joined → True, PlotRange → All,
 FrameLabel -> {"Date", "Energy prices $/MWh"}, LabelStyle → {16},
 PlotLabel → Style["Energy Prices in California", 20, Black],
 DateTicksFormat -> {"Day", "/", "MonthShort", "/", "YearShort"}]
```

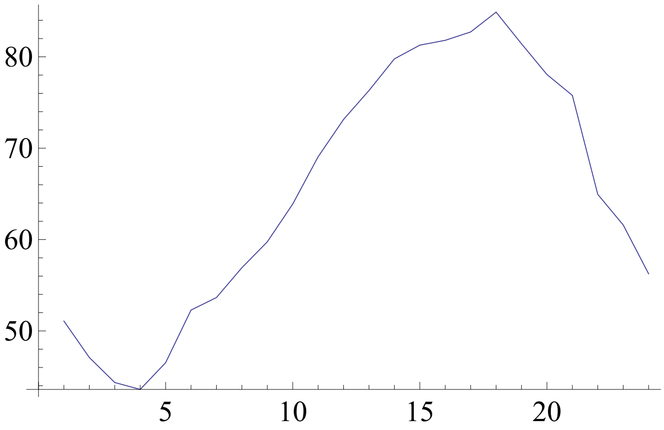


```
Length[pricehourly]
```

```
24 888
```

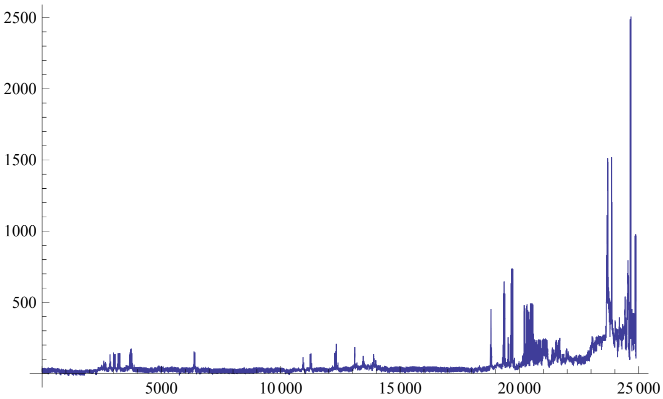
```
day = Map[Mean[#] &, Transpose[Partition[pricehourly, 24]]];
```

```
ListPlot[day, Joined → True,
 FrameLabel -> {"Hour", "Average energy prices $/MWh"}, LabelStyle → {16}]
```



```
v = Flatten[Table[day, {1037}]];
ListPlot[v, Joined → True, PlotRange → All];
```

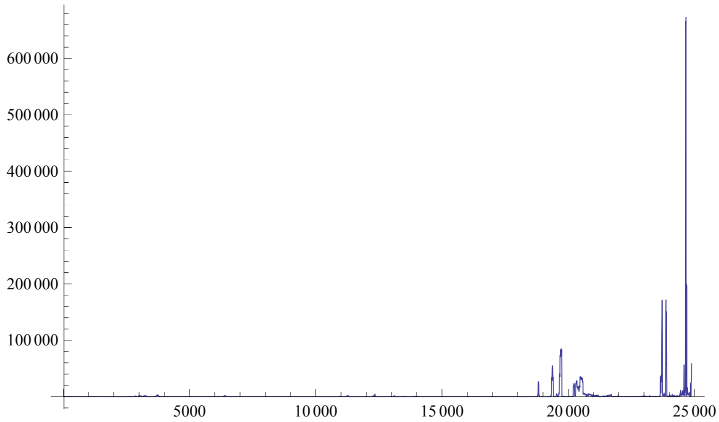
```
X = pricehourly - v + Mean[v];
ListPlot[X, Joined → True, PlotRange → All]
```



```
t = 50;
Variance[X[[t - 12 ;; t]]]
```

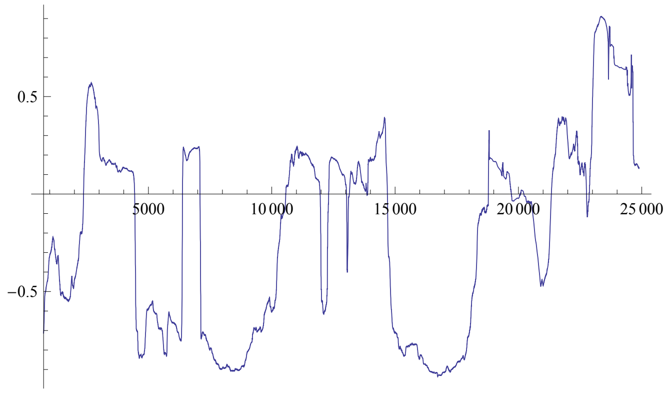
```
114.956
```

```
variancelist = Table[{t, Variance[X[[t - 24 ;; t]]]}, {t, 25, 24 888}];
ListPlot[variancelist, Joined -> True, PlotRange → All]
```



```
n = 12;
corlist = Table[
   {t, Correlation[Drop[X[[t - 30 * 24 ;; t]], n], Drop[X[[t - 30 * 24 ;; t]], -n]]},
   {t, 25 * 30, 24 888}];
```

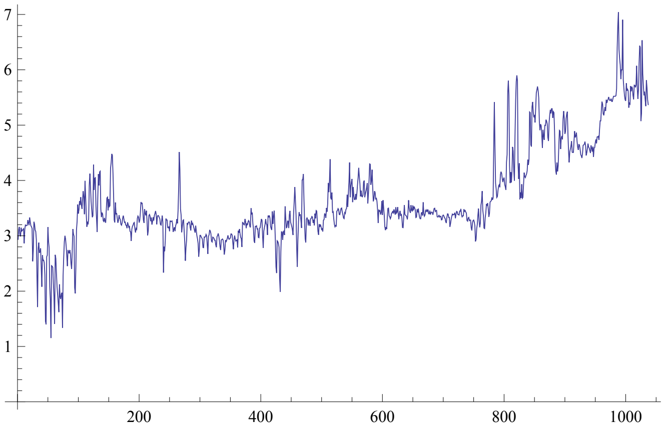
```
ListPlot[corlist, Joined -> True, PlotRange → All]
```



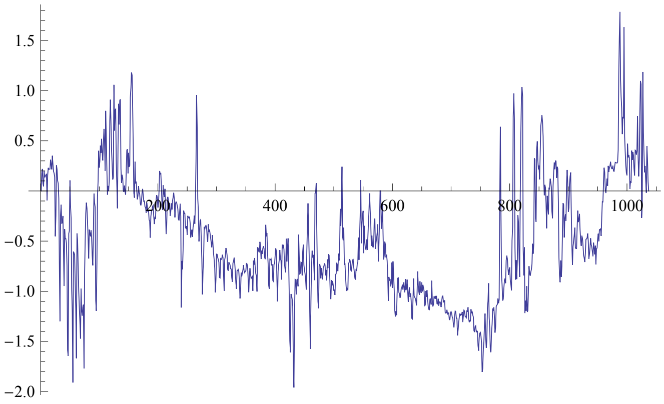
```
pars = FindProcessParameters[Log[Y], FractionalBrownianMotionProcess[h]]
```

```
Y = Map[Mean[#] &, Partition[pricehourly, 24]];
```

```
ListPlot[Log[Y], Joined -> True]
```



```
ΔY = Drop[Log[Y], 1] - Drop[Log[Y], -1];
Z = FoldList[Plus, 0, ΔY - Mean[ΔY]];
ListPlot[Z, Joined -> True]
```



```
Length[Z]
1037
```

We have 1037 day - long sequence of average daily prices. We divide the sample into subsamples of equal lenght. Since the number 1037 has only two divisors greater 10 we analyze a sample of length 1036, which has 7 divisors greater 10: 14, 28, 37, 74, 148, 259, 518.

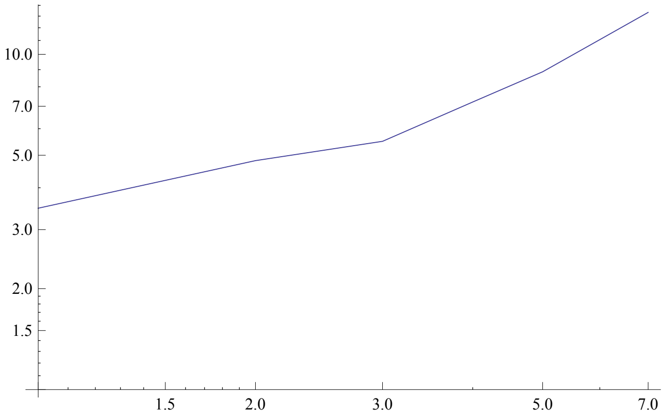
```
Divisors[1037]
{1, 17, 61, 1037}

d = Divisors[1036]
{1, 2, 4, 7, 14, 28, 37, 74, 148, 259, 518, 1036}

d1 = d[[5 ;; 11]]
{14, 28, 37, 74, 148, 259, 518}

Z1 = ΔY[[1 ;; 1036]];
```

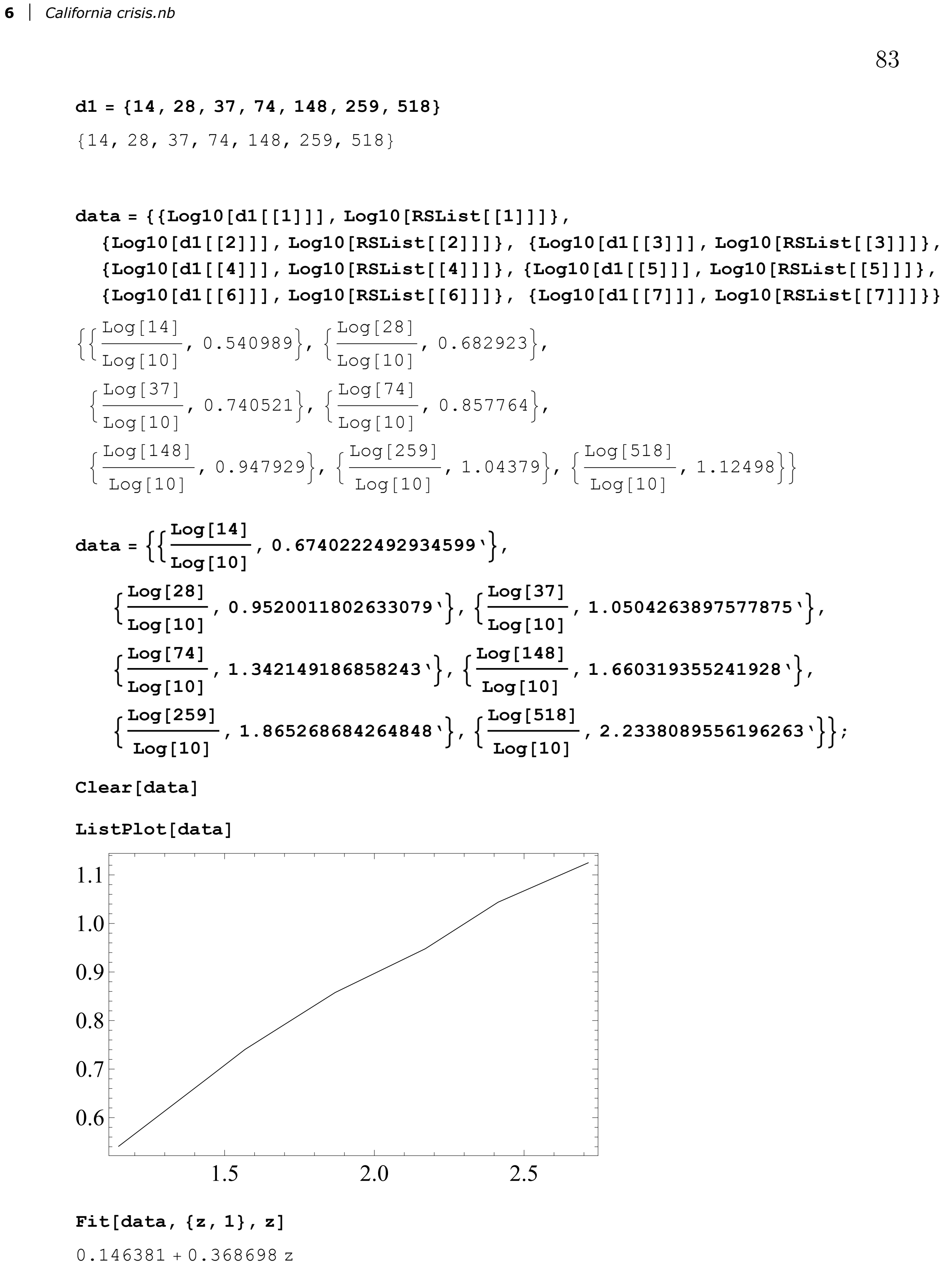
Function of Rescaled range analysis

```
RS[subsample_] := Module[{list = subsample, E, Z, Y, R, S, i, j},
  E = Mean[list];
  Z = {};
  For [i = 1, i ≤ Length[list], i++,
   AppendTo[Z, list[[i]] - E];
  ];
```

$$S = \sqrt{\frac{1}{\text{Length[list]}} \sum_{i=1}^{\text{Length[list]}} \text{Z[[i]]}^2} ;$$

```
  Y = {};
  Y = {Z[[1]]};
  For[j = 1, j < Length[list], j++,
   AppendTo[Y, Y[[j]] + Z[[j + 1]]];
  ];
  R = Max[Y] - Min[Y];
  R / S
 ]

NumberOfSubsample = Length[d1];
RSList = {};
For[i = 1, i ≤ NumberOfSubsample, i++,
 L = d1[[i]];
 n = Length[Z1] / L;
 RSsum = 0;
 For[t = 0, t < n, t++,
  subsample = Take[Z1, {t * L + 1, (t + 1) * L}];
  RSsum = RSsum + RS[subsample] / n;
 ];
 AppendTo[RSList, RSsum];
]

RSList

{3.47527, 4.81863, 5.502, 7.20715, 8.87012, 11.061, 13.3347}

ListLogLogPlot[RSList, Joined → True]
```

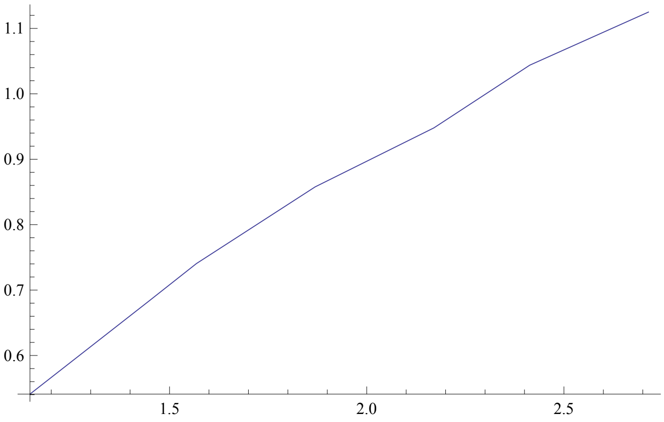




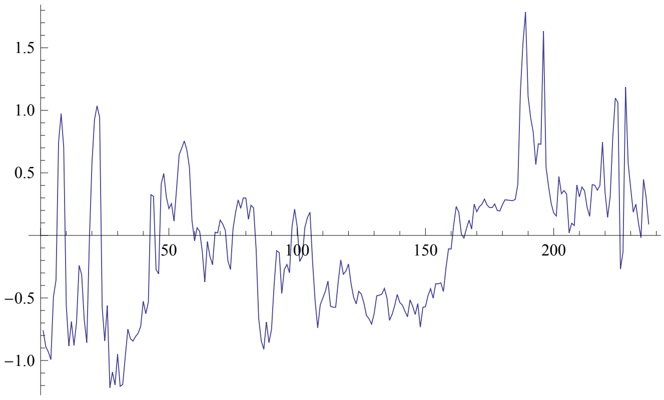
```
d1 = {14, 28, 37, 74, 148, 259, 518}

{14, 28, 37, 74, 148, 259, 518}


data = {{Log10[d1[[1]]], Log10[RSList[[1]]]},
  {Log10[d1[[2]]], Log10[RSList[[2]]]}, {Log10[d1[[3]]], Log10[RSList[[3]]]},
  {Log10[d1[[4]]], Log10[RSList[[4]]]}, {Log10[d1[[5]]], Log10[RSList[[5]]]},
  {Log10[d1[[6]]], Log10[RSList[[6]]]}, {Log10[d1[[7]]], Log10[RSList[[7]]]}}

{{Log[14]/Log[10], 0.540989}, {Log[28]/Log[10], 0.682923},
 {Log[37]/Log[10], 0.740521}, {Log[74]/Log[10], 0.857764},
 {Log[148]/Log[10], 0.947929}, {Log[259]/Log[10], 1.04379}, {Log[518]/Log[10], 1.12498}}

data = {{Log[14]/Log[10], 0.6740222492934599`},
   {Log[28]/Log[10], 0.9520011802633079`}, {Log[37]/Log[10], 1.0504263897577875`},
   {Log[74]/Log[10], 1.342149186858243`}, {Log[148]/Log[10], 1.660319355241928`},
   {Log[259]/Log[10], 1.865268684264848`}, {Log[518]/Log[10], 2.2338089556196263`}};

Clear[data]

ListPlot[data]
```

```
Fit[data, {z, 1}, z]

0.146381 + 0.368698 z
```

```
ListPlot[data, Joined → True, FrameLabel -> {"Log(d)", "Log (R/S)"}]
```



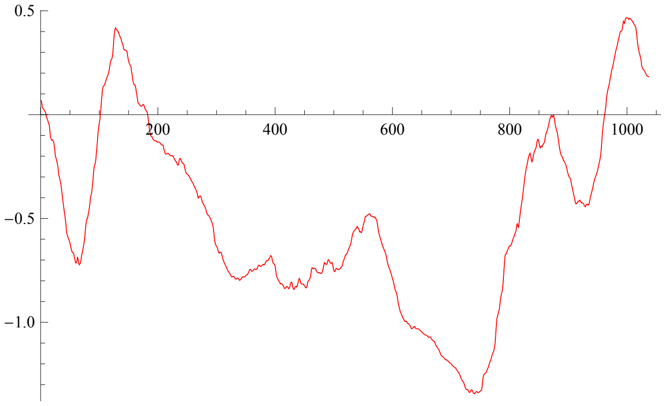
```
ListPlot[Z[[800 ;; 1036]],  Joined -> True]
```



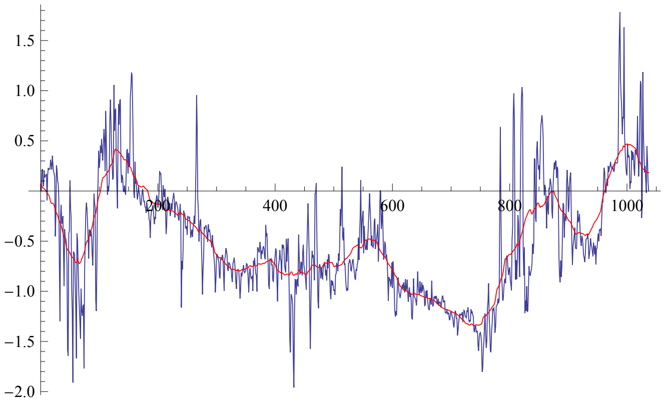
We have the moving average, Z is the time series with fixed time step Δt = 1.

```
av = Drop[MovingAverage[ArrayPad[Z, 5 * 12 / 2, "Fixed"], 5 * 12], 1];
```

```
PL2 = ListPlot[av, PlotStyle → Red, Joined → True]
PL1 = ListPlot[Z, Joined → True];
```



```
Show[PL1, PL2]
```



Appendix C

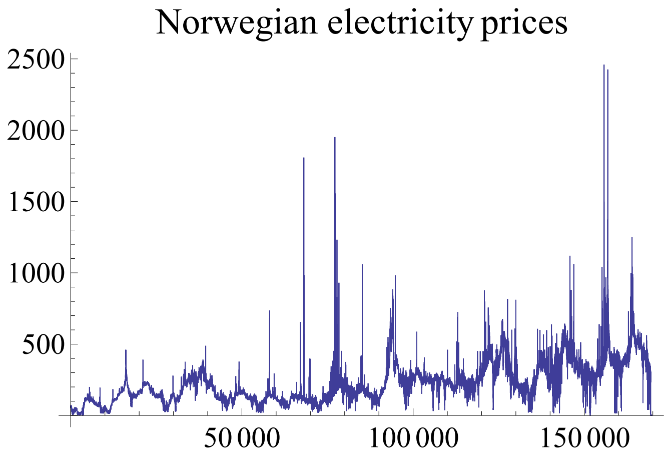
Source code of Norwegian energy prices

This appendix contains analysis of Norwegian electricity spot market. The code plots average electricity prices per day and per hour in Norway for 1992-2011 years, variance and correlation function. Also we find the Hurst exponent.

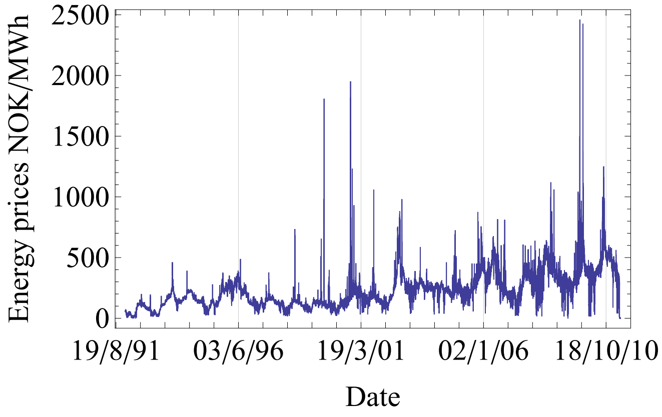
```
x =
 ToExpression[StringSplit[ReadList["C:/Users/i/Desktop/California electricity
       crisis/Norwegainprices_1992-2011.txt", String]][[All]]]
```

```
pricehourly = Flatten[x];
```

```
ListPlot[pricehourly, Joined → True, PlotRange → All, LabelStyle → {16},
 PlotLabel → Style["Norwegian electricity prices", 20, Black]]
```



```
DateListPlot[pricehourly, {1992, 1, 1, 1}, Joined → True, PlotRange → All,
 FrameLabel -> {"Date", "Energy prices NOK/MWh"}, LabelStyle → {16},
 DateTicksFormat -> {"Day", "/", "MonthShort", "/", "YearShort"}]
```



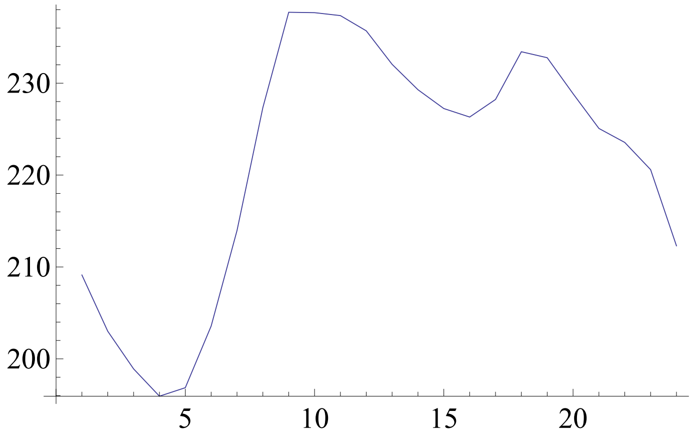
```
Length[x]
```

```
169 680
```

```
day = Map[Mean[#] &, Transpose[Partition[pricehourly, 24]]]
```

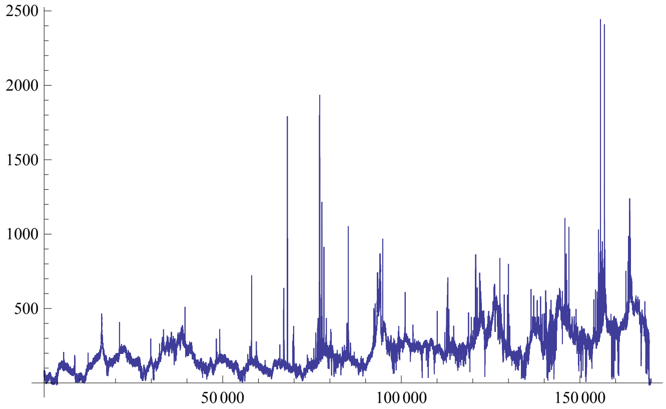
```
{209.127, 203.003, 198.906, 195.935, 196.86, 203.57, 213.963, 227.343,
 237.719, 237.675, 237.356, 235.699, 232.053, 229.29, 227.241, 226.316,
 228.227, 233.424, 232.773, 228.859, 225.076, 223.553, 220.593, 212.301}
```

```
ListPlot[day, Joined → True,
 FrameLabel -> {"Hour", "Average energy prices NOK/MWh"}, LabelStyle → {16}]
```



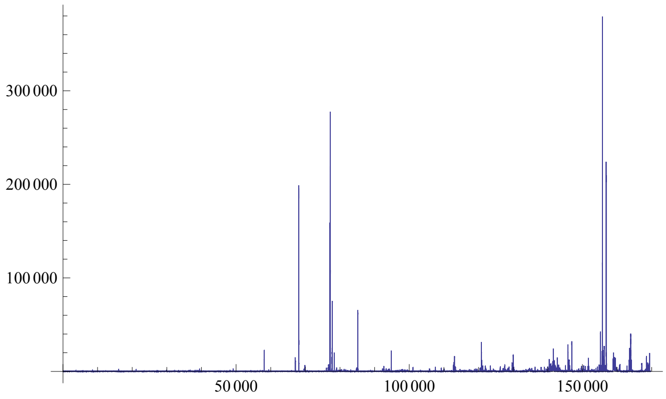
```
v = Flatten[Table[day, {7070}]];
```

```
X = pricehourly - v + Mean[v];
ListPlot[X, Joined → True, PlotRange → All]
```



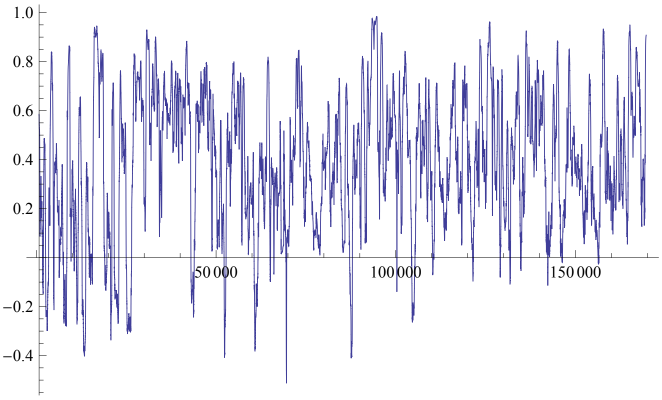
```
t = 50;
```

```
variancelist = Table[{t, Variance[X[[t - 24 ;; t]]]}, {t, 25, 169 680}];
ListPlot[variancelist, Joined -> True, PlotRange → All]
```



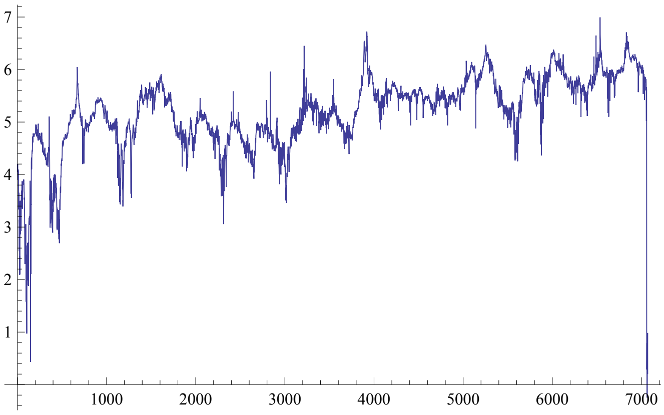
```
n = 12;
corlist = Table[
   {t, Correlation[Drop[X[[t - 30 * 24 ;; t]], n], Drop[X[[t - 30 * 24 ;; t]], -n]]},
   {t, 25 * 30, 169 680}];
```

```
ListPlot[corlist, Joined -> True, PlotRange → All]
```

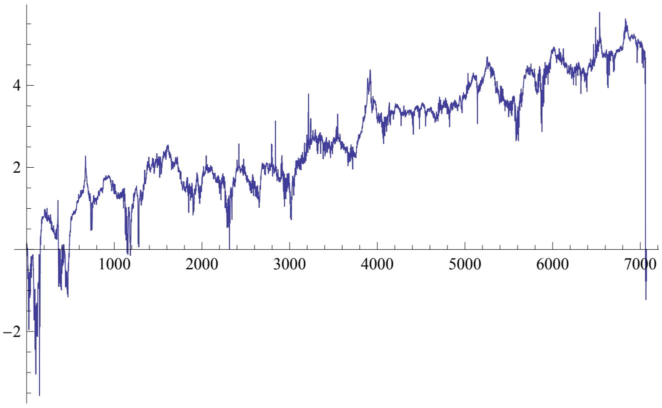


```
Y = Map[Mean[#] &, Partition[pricehourly, 24]];
```

```
ListPlot[Log[Y], Joined -> True, PlotRange → All]
```



```
ΔY = Drop[Log[Y], 1] - Drop[Log[Y], -1];
Z = FoldList[Plus, 0, ΔY - Mean[ΔY]];
ListPlot[Z, Joined -> True]
```



```
Length[ΔY]
```

```
7069
```

We have 7069 day - long sequence of average daily prices. We divide the sample into subsamples of equal lenght. Since the number 7069 doesn't have divisors we analyze a sample of length 7068, which has 18 divisors greater 10 : 12, 19, 31, 38, 57, 62, 76, 93, 114, 124, 186, 228, 372, 589, 1178, 1767, 2356, 3534.

```
Divisors[7069]
```

```
{1, 7069}
```

```
d = Divisors[7068]
```

```
{1, 2, 3, 4, 6, 12, 19, 31, 38, 57, 62, 76, 93,
 114, 124, 186, 228, 372, 589, 1178, 1767, 2356, 3534, 7068}
```

```
d1 = d[[6 ;; 23]]
```

```
{12, 19, 31, 38, 57, 62, 76, 93, 114, 124, 186, 228, 372, 589, 1178, 1767, 2356, 3534}
```

```
Z1 = ΔY[[1 ;; 7068]];
```

Function of Rescaled range analysis

```
RS[subsample_] := Module[{list = subsample, E, Z, Y, R, S, i, j},
  E = Mean[list];
  Z = {};
  For [i = 1, i ≤ Length[list], i++,
   AppendTo[Z, list[[i]] - E];
  ];
```

$$S = \sqrt{\frac{1}{\mathrm{Length[list]}} \sum_{i=1}^{\mathrm{Length[list]}} \mathrm{Z[[i]]}^2} \;;$$

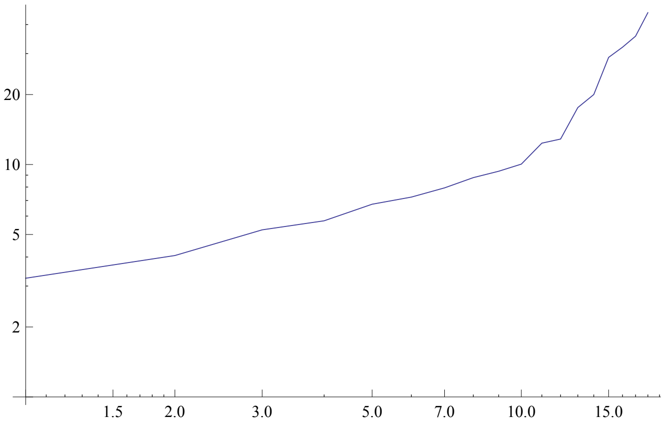
```
  Y = {};
  Y = {Z[[1]]};
  For[j = 1, j < Length[list], j++,
   AppendTo[Y, Y[[j]] + Z[[j + 1]]];
  ];
  R = Max[Y] - Min[Y];
  R / S
 ]
```

```
NumberOfSubsample = Length[d1];
RSList = {};
For[i = 1, i ≤ NumberOfSubsample, i++,
 L = d1[[i]];
 n = Length[Z1] / L;
 RSsum = 0;
 For[t = 0, t < n, t++,
  subsample = Take[Z1, {t * L + 1, (t + 1) * L}];
  RSsum = RSsum + RS[subsample] / n;
 ];
 AppendTo[RSList, RSsum];
]
```

```
RSList
```

```
{3.24166, 4.05905, 5.23513, 5.72585, 6.74822, 7.24215, 7.93451, 8.78215, 9.35453,
 10.0393, 12.3561, 12.8698, 17.5801, 19.9984, 28.8975, 31.9677, 35.655, 45.0119}
```

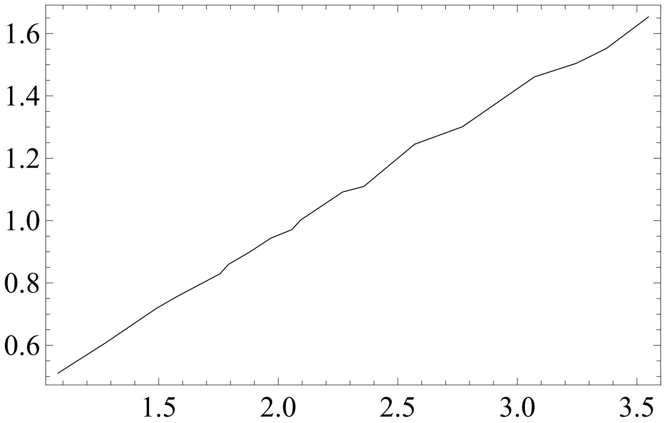
```
ListLogLogPlot[RSList, Joined → True]
```



```
data = {{Log10[d1[[1]]], Log10[RSList[[1]]]},
   {Log10[d1[[2]]], Log10[RSList[[2]]]}, {Log10[d1[[3]]], Log10[RSList[[3]]]},
   {Log10[d1[[4]]], Log10[RSList[[4]]]}, {Log10[d1[[5]]], Log10[RSList[[5]]]},
   {Log10[d1[[6]]], Log10[RSList[[6]]]}, {Log10[d1[[7]]], Log10[RSList[[7]]]},
   {Log10[d1[[8]]], Log10[RSList[[8]]]}, {Log10[d1[[9]]], Log10[RSList[[9]]]},
   {Log10[d1[[10]]], Log10[RSList[[10]]]}, {Log10[d1[[11]]],
    Log10[RSList[[11]]]}, {Log10[d1[[12]]], Log10[RSList[[12]]]},
   {Log10[d1[[13]]], Log10[RSList[[13]]]}, {Log10[d1[[14]]],
    Log10[RSList[[14]]]}, {Log10[d1[[15]]], Log10[RSList[[15]]]},
   {Log10[d1[[16]]], Log10[RSList[[16]]]}, {Log10[d1[[17]]],
    Log10[RSList[[17]]]}, {Log10[d1[[18]]], Log10[RSList[[18]]]}}
```

```
   Log[12]               Log[19]               Log[31]
{{--------, 0.510768}, {--------, 0.608425}, {--------, 0.718927},
   Log[10]               Log[10]               Log[10]

  Log[38]             Log[57]               Log[62]
 {--------, 0.75784}, {--------, 0.829189}, {--------, 0.859868},
  Log[10]             Log[10]               Log[10]

  Log[76]             Log[93]               Log[114]
 {--------, 0.89952}, {--------, 0.943601}, {--------, 0.971022},
  Log[10]             Log[10]                Log[10]

  Log[124]            Log[186]              Log[228]
 {--------, 1.0017}, {--------, 1.09188}, {--------, 1.10957},
   Log[10]             Log[10]               Log[10]

  Log[372]             Log[589]              Log[1178]
 {--------, 1.24502}, {--------, 1.30099}, {----------, 1.46086},
   Log[10]              Log[10]                Log[10]

  Log[1767]             Log[2356]              Log[3534]
 {----------, 1.50471}, {----------, 1.55212}, {----------, 1.65333}}
   Log[10]                Log[10]                Log[10]
```

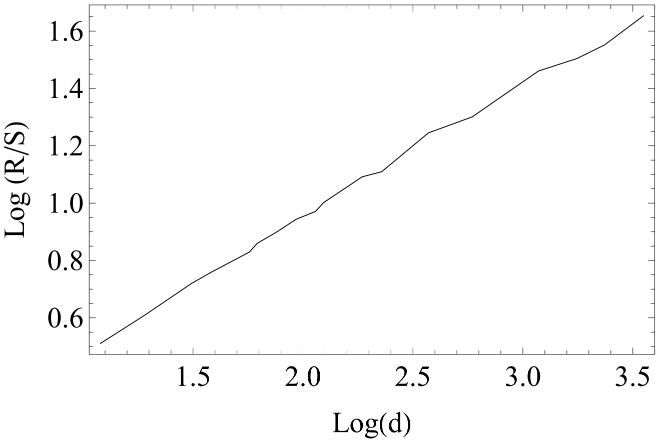
```
ListPlot[data]
```



```
Fit[data, {z, 1}, z]
```

```
0.0339938 + 0.458081 z
```

```
ListPlot[data, Joined → True, FrameLabel -> {"Log(d)", "Log (R/S)"}]
```



```
Hurst exponent is equal to 0.458081
```

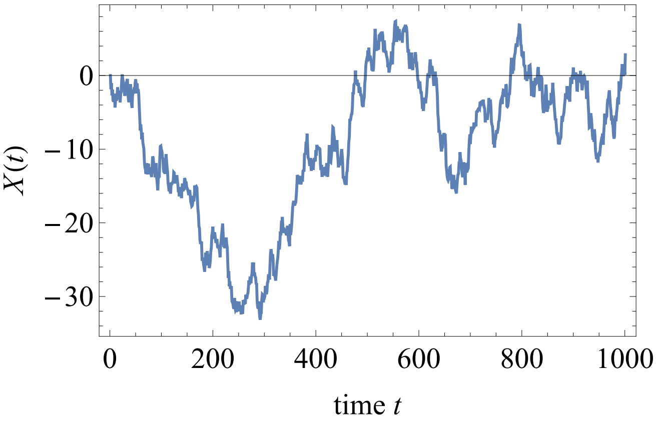
Appendix D

Source code of California early warning analysis

This part includes analysis of early warning signals for California Power Crisis. In the beginning we use the model of random walk process, then the model of fractional Brownian motion process.

Consider a random walk process:

```
n = 1000;
X = FoldList[Plus, 0, RandomReal[NormalDistribution [0, 1], n]];
PL1 = ListPlot[X, Joined → True, Frame → True,
  FrameStyle → Directive[16], FrameLabel → {"time  t", "X(t)"} ]
```

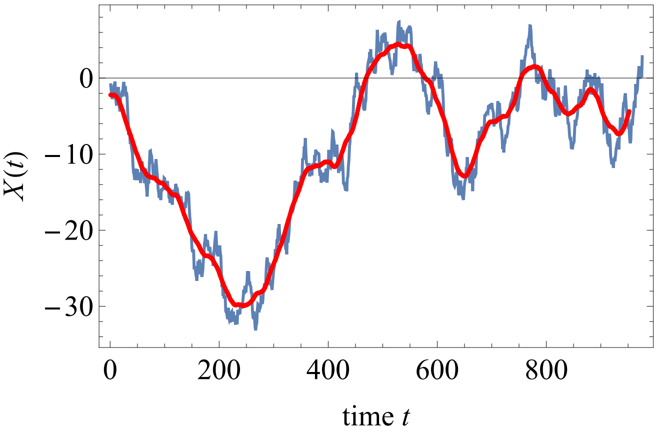


If we estimate the θ in an OU process, then we get something close to zero, because θ=0 for a Brownian motion.

```
EstimatedProcess[X, OrnsteinUhlenbeckProcess[μ, σ, θ]]
```

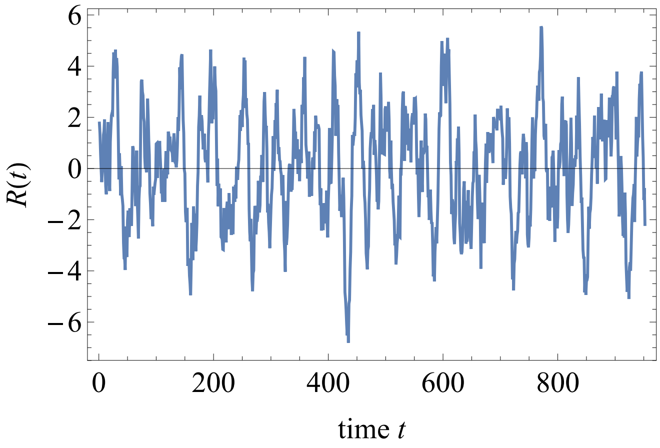
```
OrnsteinUhlenbeckProcess[-8.69676, 10.2675, 0.00454413]
```

```
win = 50;
L = MovingAverage[X, win];
PL1 = ListPlot[Drop[X, win / 2], Joined → True, Frame → True,
   FrameStyle → Directive[16], FrameLabel → {"time  t", "X(t)"} ];
PL2 = ListPlot[L, PlotStyle → Red];
residual = Drop[Drop[X, win / 2 - 1], -win / 2] - L;
Show[PL1, PL2]
```



If we make a smoothing of the signal *S(t)* (the red curve), then the residual *R(t)=X(t)-S(t)* is well described by an OU process with θ corresponding to the time scale in the smoothing.

```
PL1 = ListPlot[residual, Joined → True, Frame → True,
  FrameStyle → Directive[16], FrameLabel → {"time t", "R(t)"} ]
```

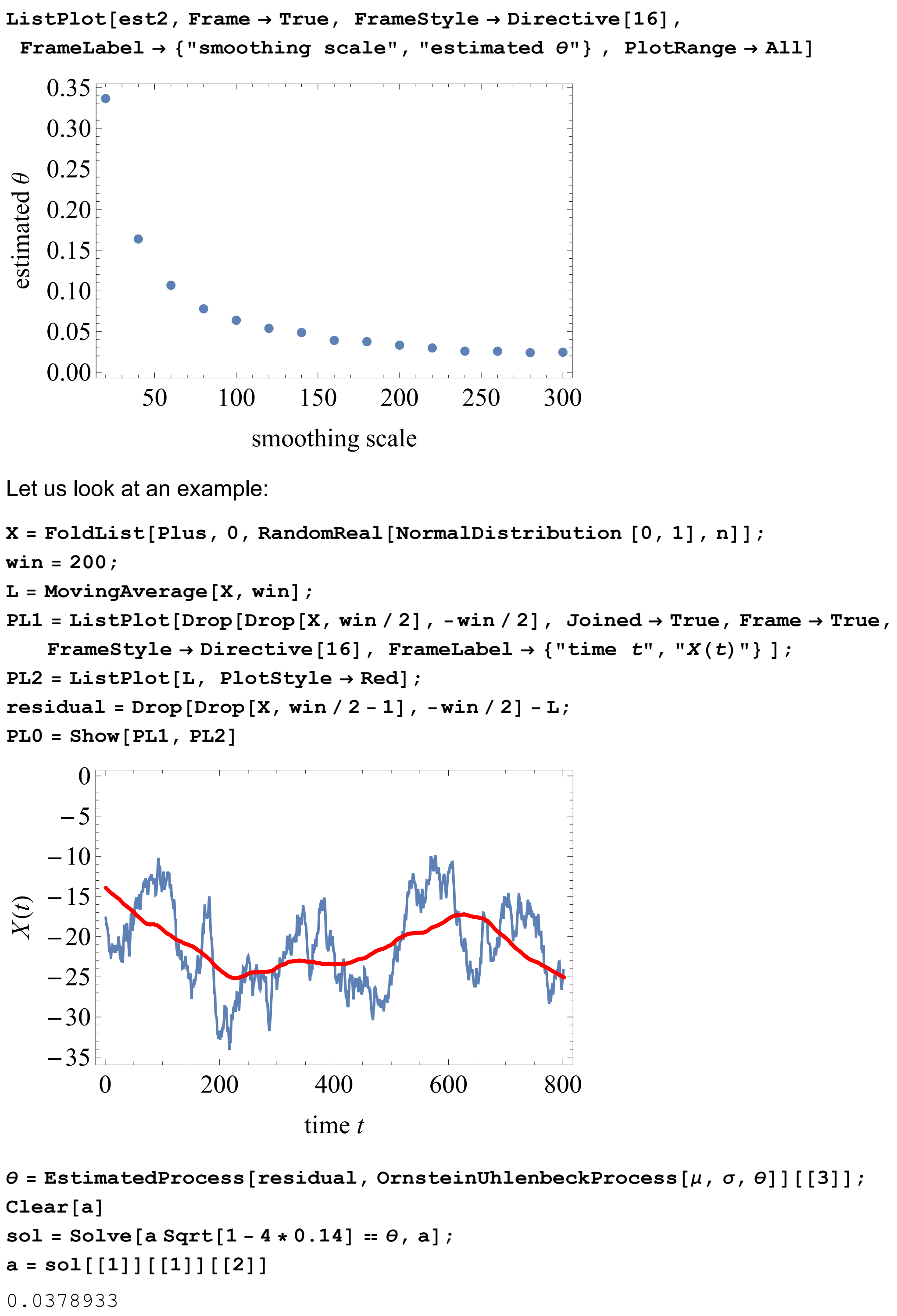


```
Clear[μ, σ]
EstimatedProcess[residual, OrnsteinUhlenbeckProcess[μ, σ, θ]]
```

```
OrnsteinUhlenbeckProcess[-0.0393581, 2.27934, 0.103088]
```

Let us look at this relationship:

```
Monitor[
 est2 = {};
 Do[
  est1 = {};
  Do[
   win = 2 * w;
   X = FoldList[Plus, 0, RandomReal[NormalDistribution [0, 1], n]];
   L = MovingAverage[X, win];
   residual = Drop[Drop[X, win / 2 - 1], -win / 2] - L;
   est1 = Append[est1,
     EstimatedProcess[residual, OrnsteinUhlenbeckProcess[μ, σ, θ]][[3]]];
   , {i, 1, 40}];
  est2 = Append[est2, {win, Mean[est1]}];
  , {w, 10, 150, 10}];
 , win]
```

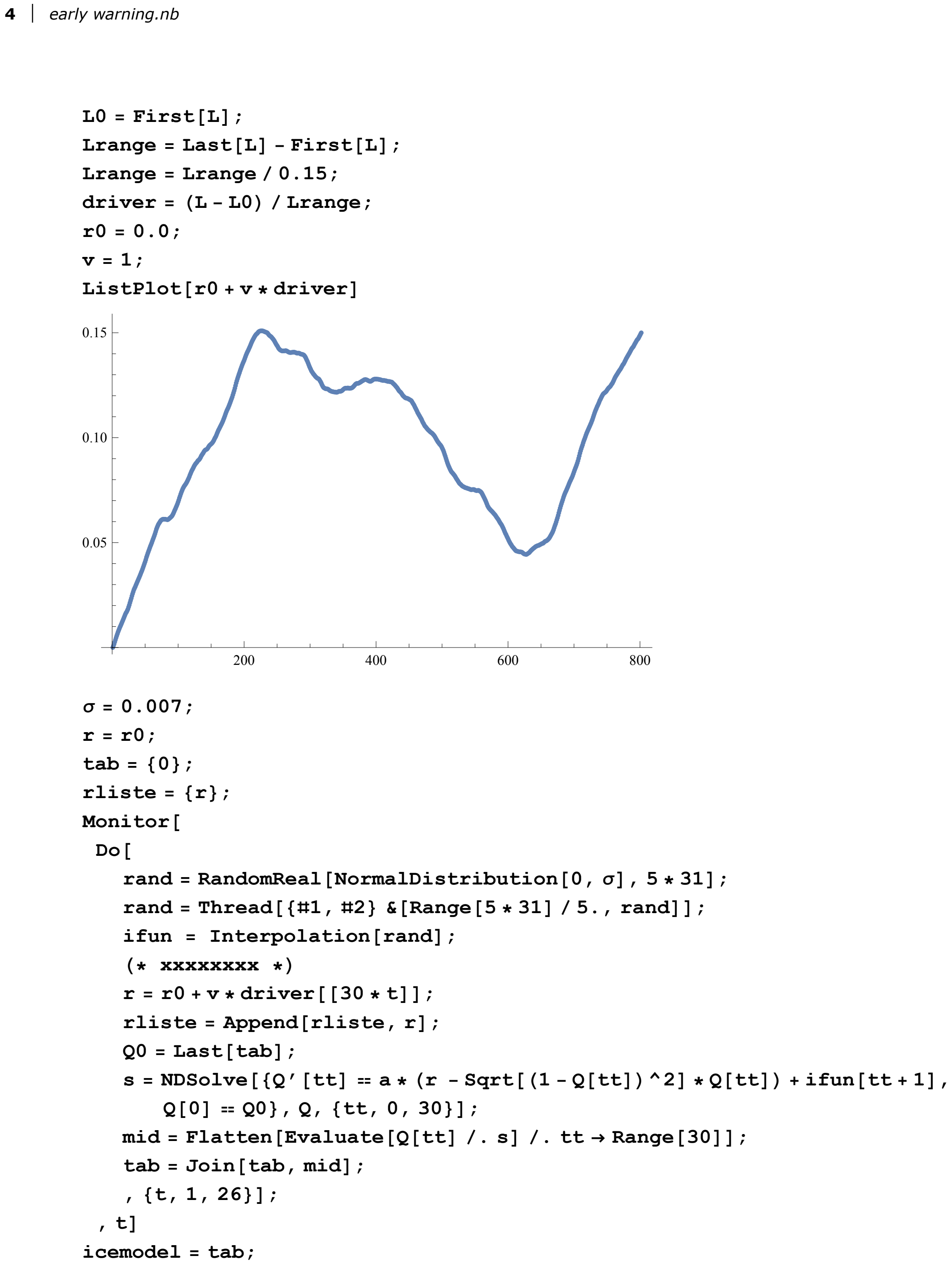


```
ListPlot[est2, Frame → True, FrameStyle → Directive[16],
 FrameLabel → {"smoothing scale", "estimated θ"} , PlotRange → All]
```

Let us look at an example:

```
X = FoldList[Plus, 0, RandomReal[NormalDistribution [0, 1], n]];
win = 200;
L = MovingAverage[X, win];
PL1 = ListPlot[Drop[Drop[X, win / 2], -win / 2], Joined → True, Frame → True,
    FrameStyle → Directive[16], FrameLabel → {"time t", "X(t)"} ];
PL2 = ListPlot[L, PlotStyle → Red];
residual = Drop[Drop[X, win / 2 - 1], -win / 2] - L;
PL0 = Show[PL1, PL2]
```

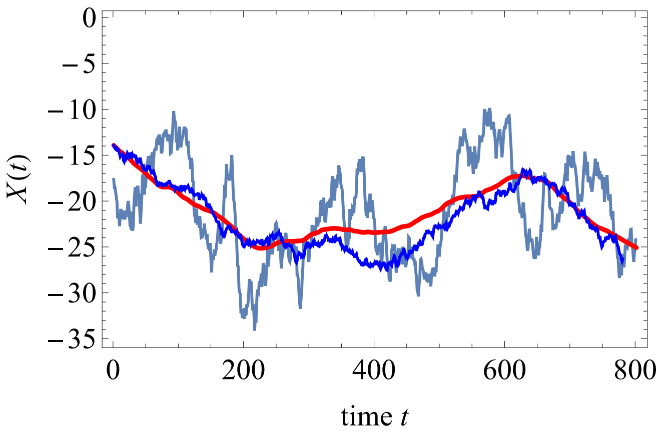
```
θ = EstimatedProcess[residual, OrnsteinUhlenbeckProcess[μ, σ, θ]][[3]];
Clear[a]
sol = Solve[a Sqrt[1 - 4 * 0.14] == θ, a];
a = sol[[1]][[1]][[2]]
0.0378933
```



```
L0 = First[L];
Lrange = Last[L] - First[L];
Lrange = Lrange / 0.15;
driver = (L - L0) / Lrange;
r0 = 0.0;
v = 1;
ListPlot[r0 + v * driver]
```

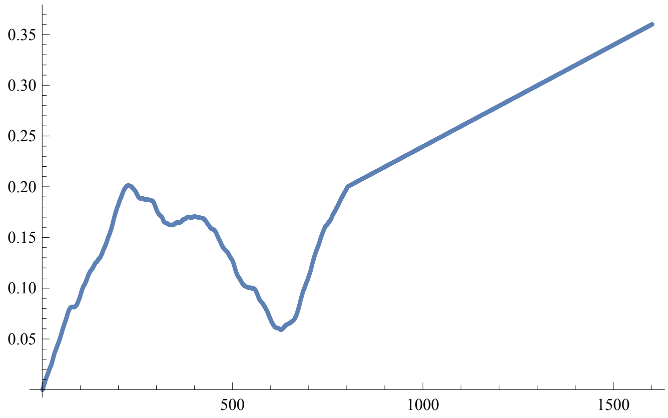
```
σ = 0.007;
r = r0;
tab = {0};
rliste = {r};
Monitor[
 Do[
   rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
   rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
   ifun  =  Interpolation[rand];
   (*  xxxxxxxx  *)
   r = r0 + v * driver[[30 * t]];
   rliste = Append[rliste, r];
   Q0 = Last[tab];
   s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt])^2] * Q[tt]) + ifun[tt + 1],
      Q[0] == Q0}, Q, {tt, 0, 30}];
   mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
   tab = Join[tab, mid];
   , {t, 1, 26}];
 , t]
icemodel = tab;
```

```
Q2 = ListPlot[L0 + Lrange * icemodel,
   Joined → True, PlotRange → All, PlotStyle → Blue];
Show[{PL0, Q2}, PlotRange → All]
```



Now we can construct a financial crash by going to the tipping point:

```
L0 = First[L];
Lrange = Last[L] - First[L];
Lrange = Lrange / 0.20;
driver = (L - L0) / Lrange;
driver = Join[driver, Last[driver] + 0.0002 * Range[800]];
ListPlot[r0 + v * driver]
```

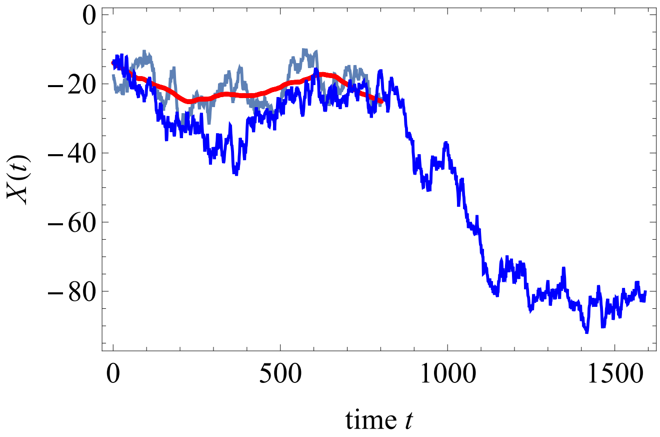


```
Length[driver] / 30.
```

```
53.4
```

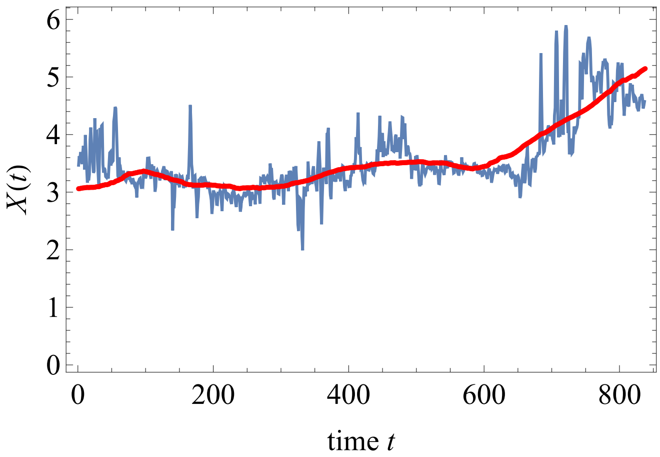
```
σ = 0.05;
r = r0;
tab = {0};
rliste = {r};
Monitor[
 Do[
   rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
   rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
   ifun  = Interpolation[rand];
   (* xxxxxxxx *)
   r = r0 + v * driver[[30 * t]];
   rliste = Append[rliste, r];
   Q0 = Last[tab];
   s = NDSolve[{Q'[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
      Q[0] == Q0}, Q, {tt, 0, 30}];
   mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
   tab = Join[tab, mid];
   , {t, 1, 53}];
 , t]
icemodel = tab;
```

```
Q2 = ListPlot[L0 + Lrange * icemodel,
   Joined → True, PlotRange → All, PlotStyle → Blue];
Show[{PL0, Q2}, PlotRange → All]
```



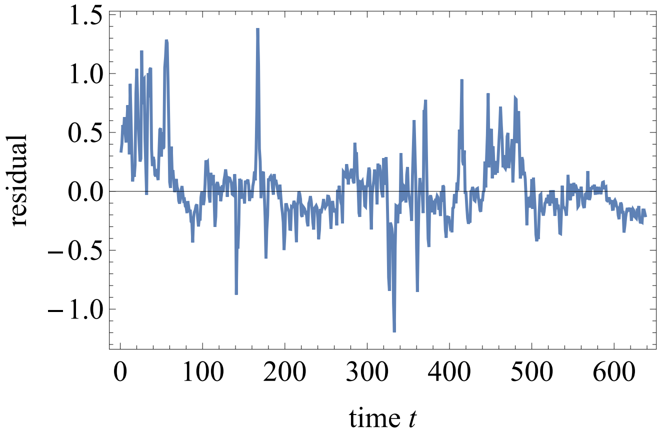
Now we look at the Power data :

```
X = ReadList["Dropbox/Master Theses/Calpx.txt"];
X = Map[Mean[#] &, Partition[X, 24]];
X = Log[X];
win = 200;
L = MovingAverage[X, win];
PL1 = ListPlot[Drop[Drop[X, win / 2], -win / 2], Joined → True, Frame → True,
   FrameStyle → Directive[16], FrameLabel → {"time  t", "X(t)"} ];
PL2 = ListPlot[L, PlotStyle → Red];
residual = Drop[Drop[X, win / 2 - 1], -win / 2] - L;
PL0 = Show[PL1, PL2]
```



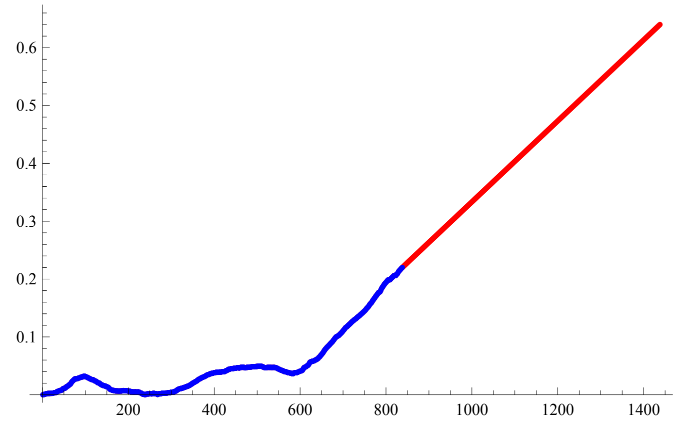
Residual before the crisis:

```
ListPlot[Drop[residual, -200], Joined → True,
 Frame → True, FrameStyle → Directive[16],
 FrameLabel → {"time  t", "residual"}, PlotRange → All ]
```



```
θ = EstimatedProcess[Drop[residual, -200],
    OrnsteinUhlenbeckProcess[μ, σ, θ]][[3]];
Clear[a]
sol = Solve[a Sqrt[1 - 4 * 0.14] == θ, a];
a = sol[[1]][[1]][[2]]
0.0378933
```

```
L0 = First[L];
Lrange = Last[L] - First[L];
Lrange = Lrange / 0.22;
driver = (L - L0) / Lrange;
r0 = 0.0;
v = 1;
PL0 = ListPlot[r0 + v * driver, PlotStyle → Blue];
driver = Join[driver, Last[driver] + 0.0007 * Range[600]];
PL1 = ListPlot[r0 + v * driver, PlotStyle → Red];
Show[PL1, PL0]
```

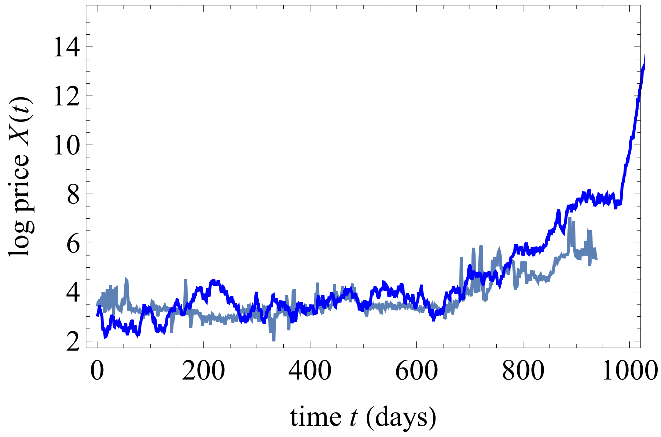


```
Length[driver] / 30.
```

```
47.9333
```

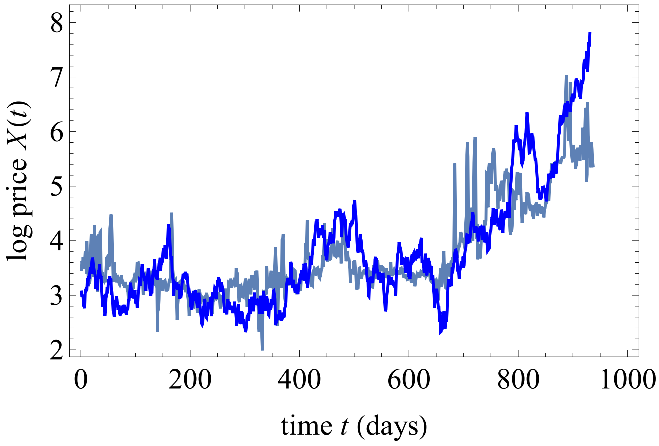
```
σ = 0.03;
r = r0;
tab = {0};
rliste = {r};
Monitor[
 Do[
   rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
   rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
   ifun  = Interpolation[rand];
   (* xxxxxxxx *)
   r = r0 + v * driver[[30 * t]];
   rliste = Append[rliste, r];
   Q0 = Last[tab];
   s = NDSolve[{Q'[tt] == a * (r - Sqrt[(1 - Q[tt])^2] * Q[tt]) + ifun[tt + 1],
      Q[0] == Q0}, Q, {tt, 0, 30}];
   mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
   tab = Join[tab, mid];
   , {t, 1, 47}];
 , t]
icemodel = tab;
```

```
PL1 = ListPlot[Drop[Drop[X, win / 2], 0],
   Joined → True, Frame → True, FrameStyle → Directive[16],
   FrameLabel → {"time  t  (days)", "log price X(t)"} ];
Q2 = ListPlot[L0 + Lrange * icemodel, Joined → True,
   PlotRange → All, PlotStyle → Blue];
Show[{PL1, Q2}, PlotRange → {{0, 1000}, {2, 15}}]
```

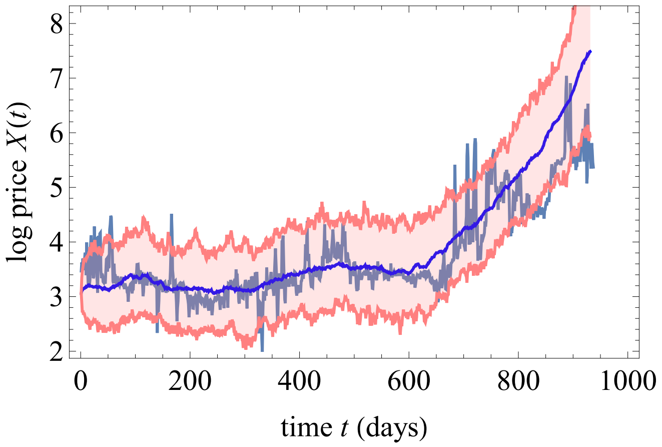


```
Z = Drop[Drop[X, win / 2], 0];
```

```
σ = 0.03;
r = r0;
tab = {0};
rliste = {r};
Monitor[
 Do[
   rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
   rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
   ifun  = Interpolation[rand];
   (* xxxxxxxx *)
   r = r0 + v * driver[[30 * t]];
   rliste = Append[rliste, r];
   Q0 = Last[tab];
   s = NDSolve[{Q'[tt] == a * (r - Sqrt[(1 - Q[tt])^2] * Q[tt]) + ifun[tt + 1],
      Q[0] == Q0}, Q, {tt, 0, 30}];
   mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
   tab = Join[tab, mid];
   , {t, 1, 31}];
 , t]
model = L0 + Lrange * tab;
PL1 = ListPlot[Z, Joined → True, Frame → True, FrameStyle → Directive[16],
   FrameLabel → {"time  t  (days)", "log price  X(t)"}];
Q2 = ListPlot[model, Joined → True, PlotRange → All, PlotStyle → Blue];
Show[{PL1, Q2}, PlotRange → {{0, 1000}, {2, 8}}]
```



```
mlist = {};
Monitor[
  Do[
    σ = 0.03;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
       rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
       ifun  =  Interpolation[rand];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q' [tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
           Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 31}];
     , t];
    model = L0 + Lrange * tab;
    mlist = Append[mlist, model];
    , {j, 1, 100}];
  , j];


mean = Map[Mean[#] &, Transpose[mlist]];
low = Map[Quantile[#, 0.025] &, Transpose[mlist]];
high = Map[Quantile[#, 1 - 0.025] &, Transpose[mlist]];

PL1 = ListPlot[Z, Joined → True, Frame → True, FrameStyle → Directive[16],
   FrameLabel → {"time  t  (days)", "log price  X(t)"} ];
Q3 = ListPlot[mean, Joined → True, PlotRange → All, PlotStyle → Blue];
Q4 = ListPlot[{low, high}, Joined → True,
   PlotRange → All, PlotStyle → {Pink}, Filling → {1 → {2}}];
Show[{PL1, Q3, Q4}, PlotRange → {{0, 1000}, {2, 8}}]
```



Early warning

```
σ = 0.03;
r = r0;
tab = {0};
rliste = {r};
Monitor[
  Do[
    rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
    rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
    ifun = Interpolation[rand];
    (* xxxxxxxx *)
    r = r0 + v * driver[[30 * t]];
    rliste = Append[rliste, r];
    Q0 = Last[tab];
    s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
        Q[0] == Q0}, Q, {tt, 0, 30}];
    mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
    tab = Join[tab, mid];
    , {t, 1, 31}];
  , t];
model = L0 + Lrange * tab;
```

```
r
```

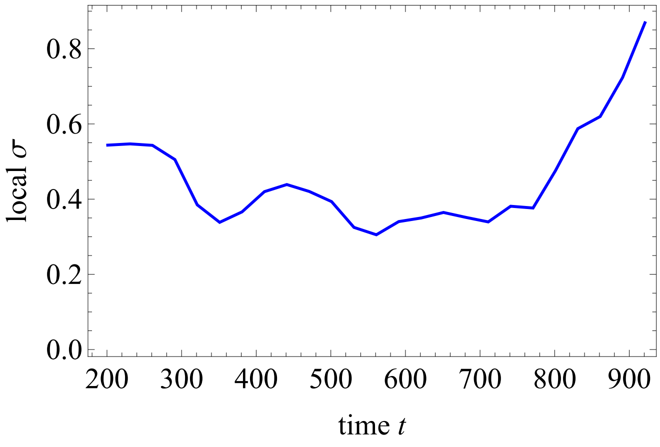
```
0.2844
```

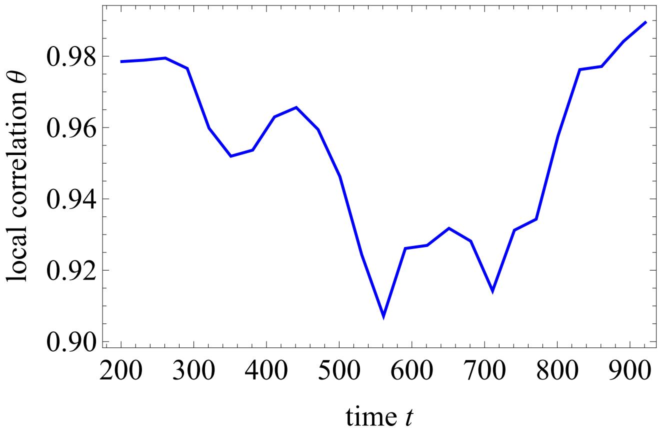
Since r<0.25 this is before the tipping point

```
θ = EstimatedProcess[model, OrnsteinUhlenbeckProcess[μ, σ, θ]][[3]]
```

```
0.0251356
```

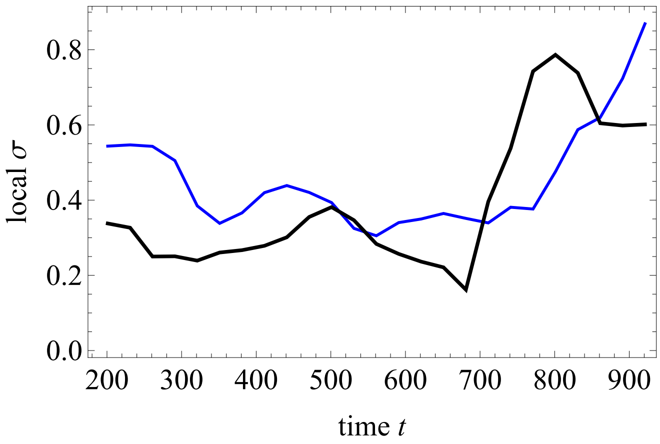
```
win = 200;
σlist = {};
τlist = {};
Monitor[
  Do[
    qqq = model[[t - win ;; t]];
    σloc = StandardDeviation[qqq]; (*local standard deviation*)
    τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
    (*local correlation scale*)
    σlist = Append[σlist, {t, σloc}];
    τlist = Append[τlist, {t, τloc}];
    , {t, win + 1, Length[model], 30}];
  , t];
QQ3 = ListPlot[σlist, Frame → True, FrameStyle → Directive[16],
    Joined → True, FrameLabel → {"time t", "local σ"}, PlotStyle → Blue];
QQ4 = ListPlot[τlist, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local correlation θ"}, PlotStyle → Blue];
Show[QQ3, PlotRange → All]
Show[QQ4, PlotRange → All]
```

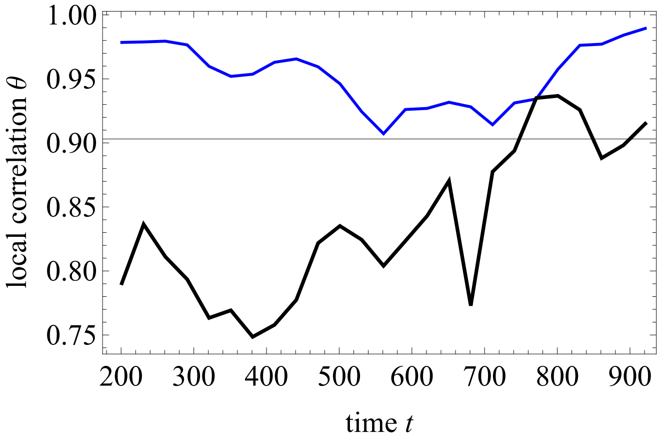


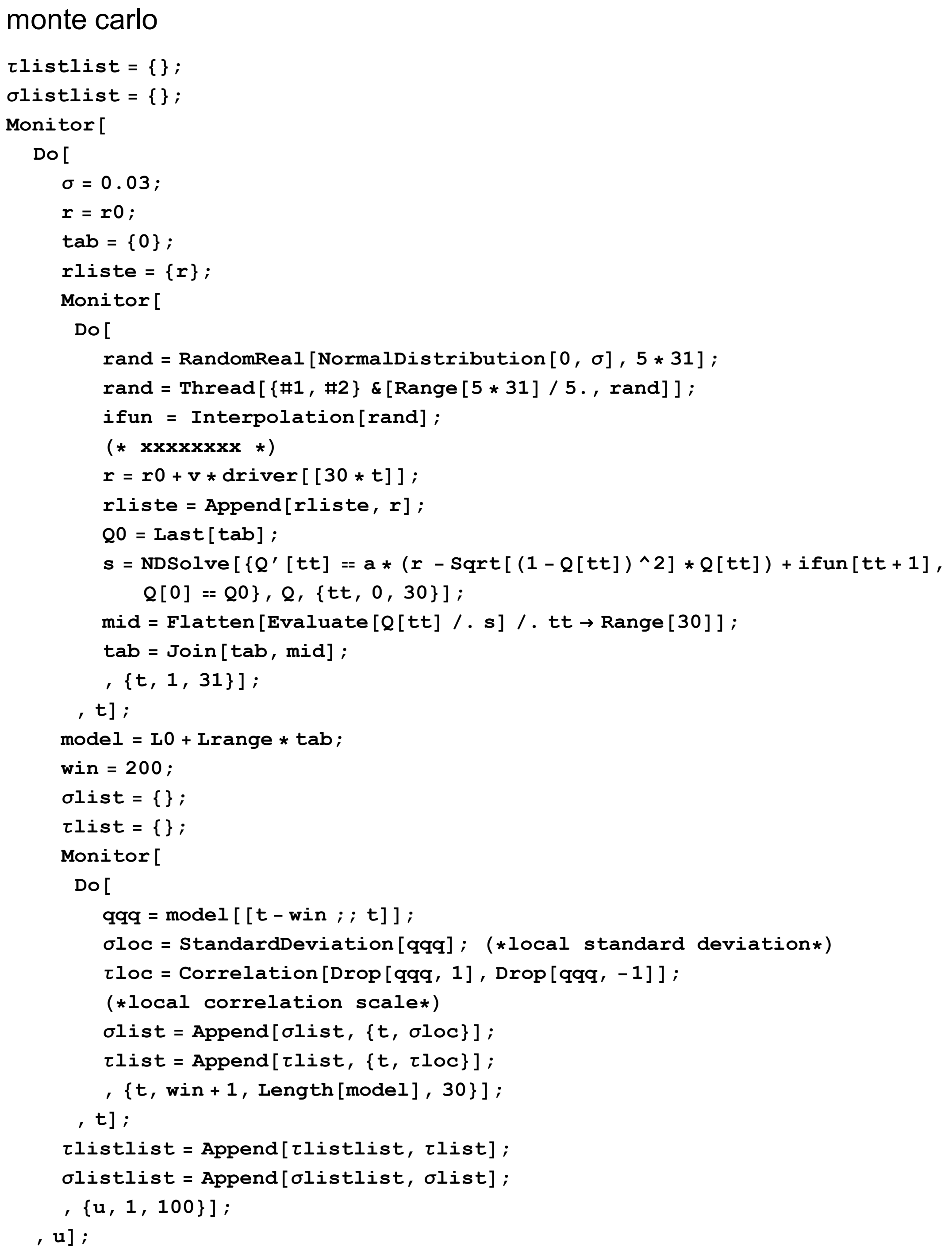


real data:

```
win = 200;
σlist = {};
τlist = {};
Monitor[
  Do[
    qqq = Z[[t - win ;; t]];
    σloc = StandardDeviation[qqq]; (*local standard deviation*)
    τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
    (*local correlation scale*)
    σlist = Append[σlist, {t, σloc}];
    τlist = Append[τlist, {t, τloc}];
    , {t, win + 1, Length[Z], 30}];
  , t];
QQ5 = ListPlot[σlist, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local σ"}, PlotStyle → {Thick, Black}];
QQ6 = ListPlot[τlist, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local correlation θ"}, PlotStyle → {Thick, Black}];
Show[{QQ3, QQ5}, PlotRange → All]
Show[{QQ4, QQ6}, PlotRange → All]
```





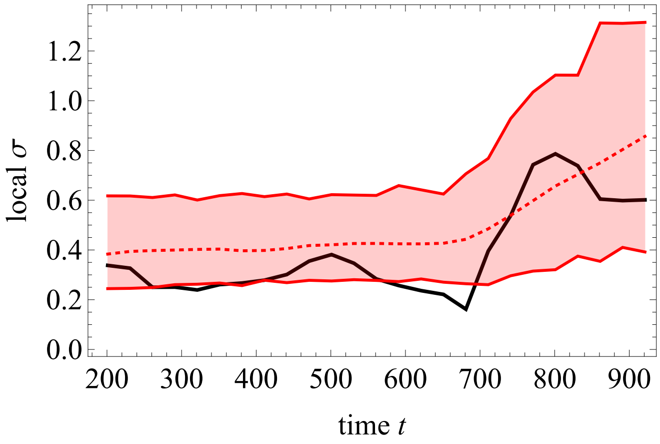


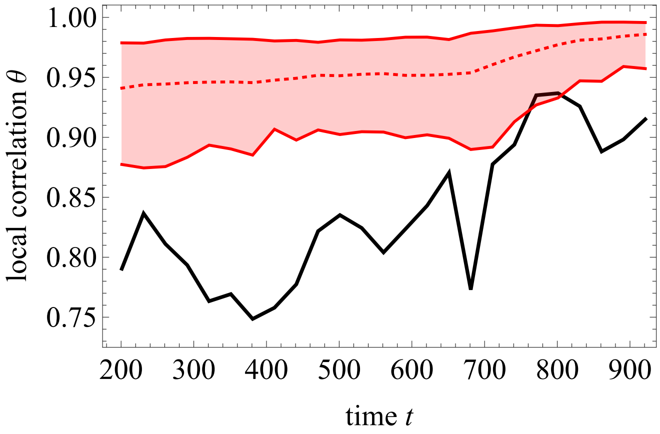
monte carlo

```
τlistlist = {};
σlistlist = {};
Monitor[
  Do[
    σ = 0.03;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
       rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
       ifun = Interpolation[rand];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
          Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 31}];
     , t];
    model = L0 + Lrange * tab;
    win = 200;
    σlist = {};
    τlist = {};
    Monitor[
     Do[
       qqq = model[[t - win ;; t]];
       σloc = StandardDeviation[qqq]; (*local standard deviation*)
       τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
       (*local correlation scale*)
       σlist = Append[σlist, {t, σloc}];
       τlist = Append[τlist, {t, τloc}];
       , {t, win + 1, Length[model], 30}];
     , t];
    τlistlist = Append[τlistlist, τlist];
    σlistlist = Append[σlistlist, σlist];
    , {u, 1, 100}];
  , u];
```

```
times = σlistlist[[1]][[All, 1]];
σmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σmean = Thread[{#1, #2} &[times, σmean]];
σlow = Thread[{#1, #2} &[times, σlow]];
σhigh = Thread[{#1, #2} &[times, σhigh]];
τmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τmean = Thread[{#1, #2} &[times, τmean]];
τlow = Thread[{#1, #2} &[times, τlow]];
τhigh = Thread[{#1, #2} &[times, τhigh]];
```

```
QQQ1 = ListPlot[{σlow, σhigh},
   Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local σ"}, PlotStyle → Red, Filling → {1 → {2}}];
QQQ2 = ListPlot[σmean, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local σ"}, PlotStyle → {Red, Dotted}];
QQQ3 = ListPlot[{τlow, τhigh}, Frame → True, FrameStyle → Directive[16],
   Joined → True, FrameLabel → {"time t", "local σ"},
   PlotStyle → Red, Filling → {1 → {2}}];
QQQ4 = ListPlot[τmean, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local σ"}, PlotStyle → {Red, Dotted}];
Show[{QQ5, QQQ1, QQQ2}, PlotRange → All]
Show[{QQ6, QQQ3, QQQ4}, PlotRange → All]
```





```
Length[Z] / 30.
```

```
31.2333
```

Local Hurst exponent

A function for de-trended fluctuation analysis:

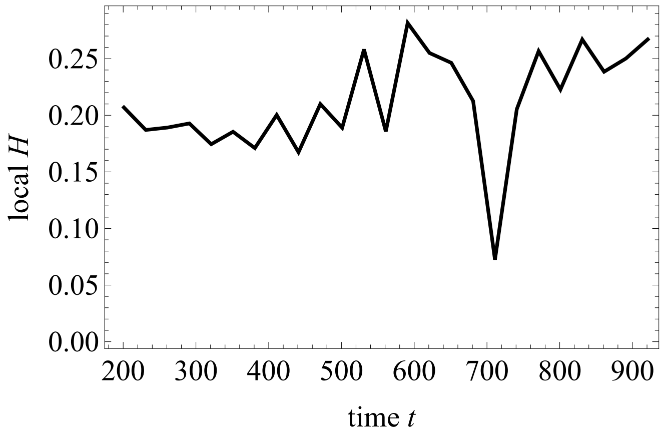
```
fit[z_, o_] := Fit[z, zz^Range[0, o], zz] /. zz → Range[Length[z]];
DFA[x_, o_, imin_, imax_] := (Module[{L, i, X, Z, var, detrends}, var = {};
    Do[L = 2^i;
     X = FoldList[Plus, 0, x - Mean[x]];
     Z = Partition[X, L];
     detrends = Map[# - fit[#, o] &, Z];
     var = Append[var, {L, Sqrt[Mean[Flatten[detrends]^2]]}];,
     {i, imin, imax}];
    Return[var];];)
```

```
Δmodel = Drop[model, 1] - Drop[model, -1];
Fit[Log[DFA[Δmodel, 1, 2, 9]], {zz, 1}, zz]
```

```
-3.25749 + 0.413678 zz
```

```
δt = 200;
t = δt + 1;
Δmodel = Drop[model, 1] - Drop[model, -1];
H = Fit[Log[DFA[Δmodel[[t - δt ;; t]], 1, 2, 7]], {zz, 1}, zz][[2]] / zz;
```

```
win = 200;
δt = win;
Hlist = {};
Monitor[
  Do[
    ΔZ = Drop[Z, 1] - Drop[Z, -1];
    qqq = ΔZ[[t - δt ;; t]];

    H = Fit[Log[DFA[qqq, 0, 2, 6]], {zz, 1}, zz][[2]] / zz;

    Hlist = Append[Hlist, {t, H}];
    , {t, win + 1, Length[Z], 30}];
  , t];
QQ3 = ListPlot[Hlist, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local H"}, PlotStyle → {Black, Thick}];
Show[QQ3, PlotRange → All]
```

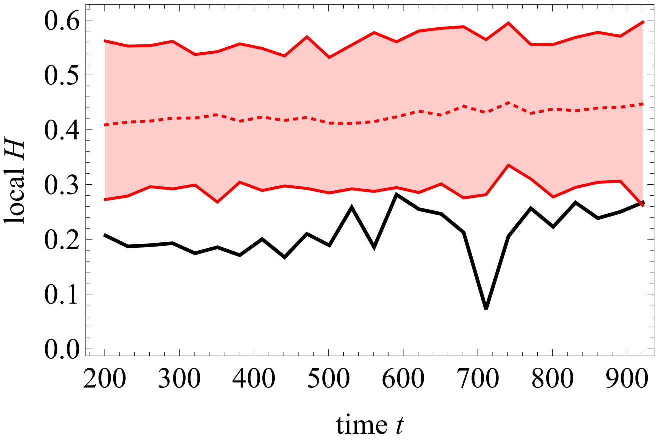


Monte Carlo for local Hurst exponent

```
Hlistlist = {};
Monitor[
  Do[
    σ = 0.03;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
       rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
       ifun = Interpolation[rand];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
           Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 31}];
     , t];
    model = L0 + Lrange * tab;
    Δmodel = Drop[model, 1] - Drop[model, -1];
    win = 200;
    δt = win;
    Hlist = {};
    Monitor[
     Do[
       qqq = Δmodel[[t - δt ;; t]];

       H = Fit[Log[DFA[qqq, 0, 2, 6]], {zz, 1}, zz][[2]] / zz;

       Hlist = Append[Hlist, {t, H}];
       , {t, win + 1, Length[Δmodel], 30}];
     , t];
    Hlistlist = Append[Hlistlist, Hlist];
    , {u, 1, 100}];
  , u];
```

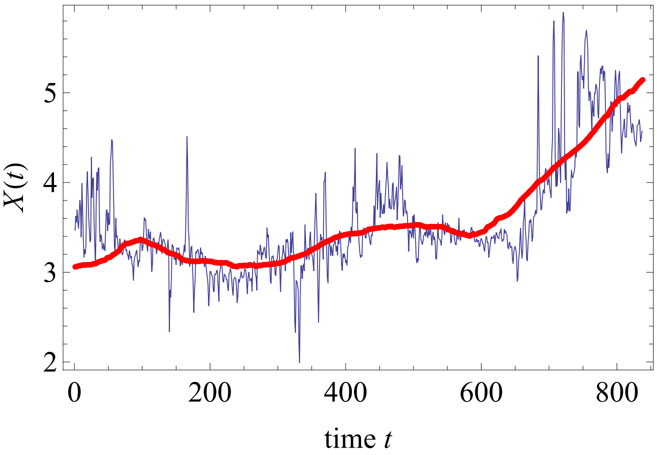
```
times = Hlistlist[[1]][[All, 1]];
Hmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hmean = Thread[{#1, #2} &[times, Hmean]];
Hlow = Thread[{#1, #2} &[times, Hlow]];
Hhigh = Thread[{#1, #2} &[times, Hhigh]];
```

```
QQ5 = ListPlot[{Hlow, Hhigh}, Frame → True, FrameStyle → Directive[16],
    Joined → True, FrameLabel → {"time t", "local \!\(\*
StyleBox[\"H\",\nFontSlant->\"Italic\"]\)"},
    PlotStyle → Red, Filling → {1 → {2}}];
QQ7 = ListPlot[Hmean, Frame → True, FrameStyle → Directive[16],
    Joined → True, FrameLabel → {"time t", "local \!\(\*
StyleBox[\"H\",\nFontSlant->\"Italic\"]\)"}, PlotStyle → {Red, Dotted}];
Show[{QQ3, QQ5, QQ7}, PlotRange → All]
```



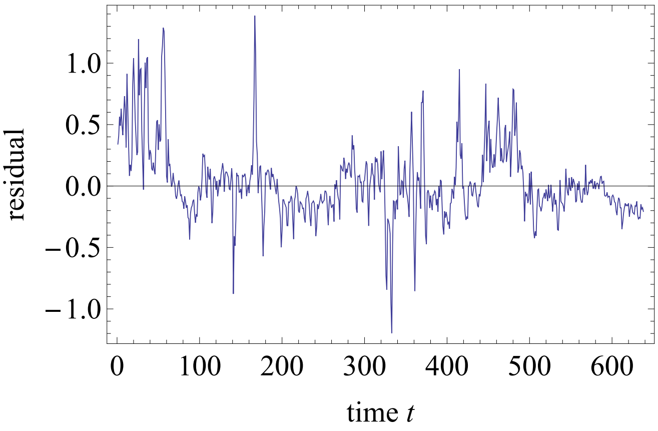
We change the white noise process to fractional Brownian motion process

```
X = ReadList["Dropbox/Master Theses/Calpx.txt"];
X = Map[Mean[#] &, Partition[X, 24]];
X = Log[X];
win = 200;
L = MovingAverage[X, win];
PL1 = ListPlot[Drop[Drop[X, win / 2], -win / 2], Joined → True, Frame → True,
   FrameStyle → Directive[16], FrameLabel → {"time t", "X(t)"} ];
PL2 = ListPlot[L, PlotStyle → Red];
residual = Drop[Drop[X, win / 2 - 1], -win / 2] - L;
PL0 = Show[PL1, PL2]
```



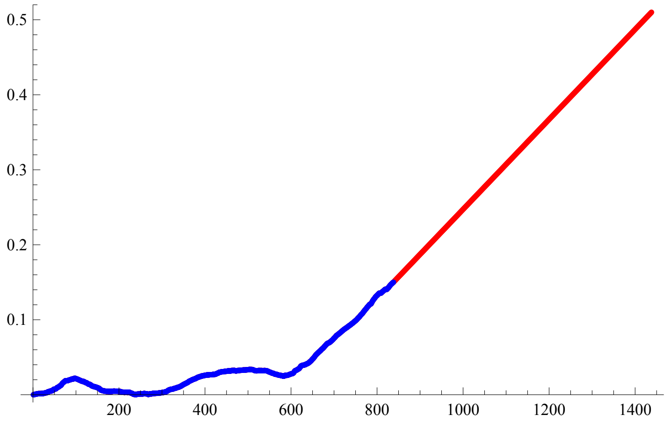
Residual before the crisis:

```
ListPlot[Drop[residual, -200], Joined → True,
 Frame → True, FrameStyle → Directive[16],
 FrameLabel → {"time t", "residual"}, PlotRange → All ]
```



```
θ = EstimatedProcess[Drop[residual, -200],
    OrnsteinUhlenbeckProcess[μ, σ, θ]][[3]];
Clear[a]
sol = Solve[a Sqrt[1 - 4 * 0.14] == θ, a];
a = sol[[1]][[1]][[2]]
0.304658
```

```
L0 = First[L];
Lrange = Last[L] - First[L];
Lrange = Lrange / 0.15;
driver = (L - L0) / Lrange;
r0 = 0.0;
v = 1;
PL0 = ListPlot[r0 + v * driver, PlotStyle → Blue];
driver = Join[driver, Last[driver] + 0.0006 * Range[600]];
PL1 = ListPlot[r0 + v * driver, PlotStyle → Red];
Show[PL1, PL0]
```

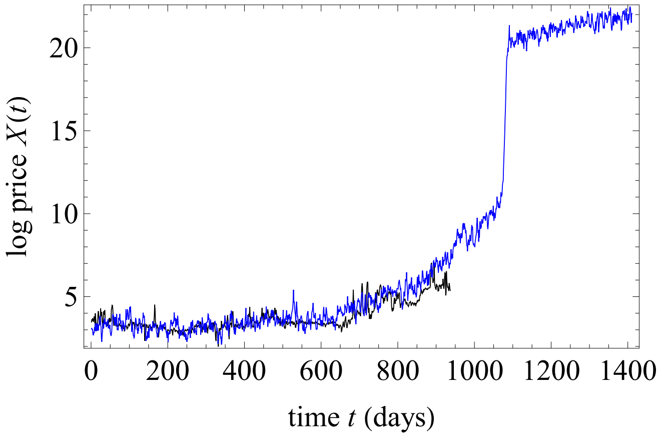


```
Length[driver] / 30.
```

```
47.9333
```

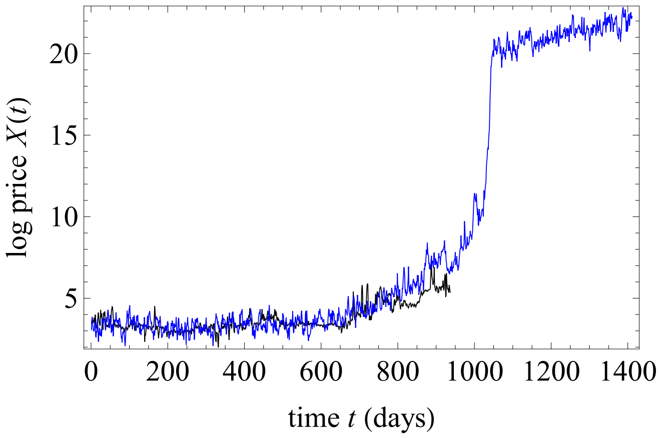
```
σ = 0.07;
r = r0;
tab = {0};
rliste = {r};
Monitor[
 Do[
   data =
    RandomFunction[FractionalBrownianMotionProcess[.368698], {0, 5 * 31 + 1, 1}];
   data = Drop[data["Path"][[All, 2]], 1];
   data = σ (Drop[data, 1] - Drop[data, -1]);
   data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
   ifun = Interpolation[data];
   (* xxxxxxxx *)
   r = r0 + v * driver[[30 * t]];
   rliste = Append[rliste, r];
   Q0 = Last[tab];
   s = NDSolve[{Q'[tt] ⩵ a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
      Q[0] ⩵ Q0}, Q, {tt, 0, 30}];
   mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
   tab = Join[tab, mid];
   , {t, 1, 47}];
 , t]
pricemodel = tab;
```

```
PL1 = ListPlot[Drop[Drop[X, win / 2], 0],
   Joined → True, Frame → True, FrameStyle → Directive[16],
   FrameLabel → {"time t (days)", "log price X(t)"}, PlotStyle → Black ];
Q2 = ListPlot[L0 + Lrange * pricemodel, Joined → True,
   PlotRange → All, PlotStyle → Blue];
Show[{PL1, Q2}, PlotRange → All]
```

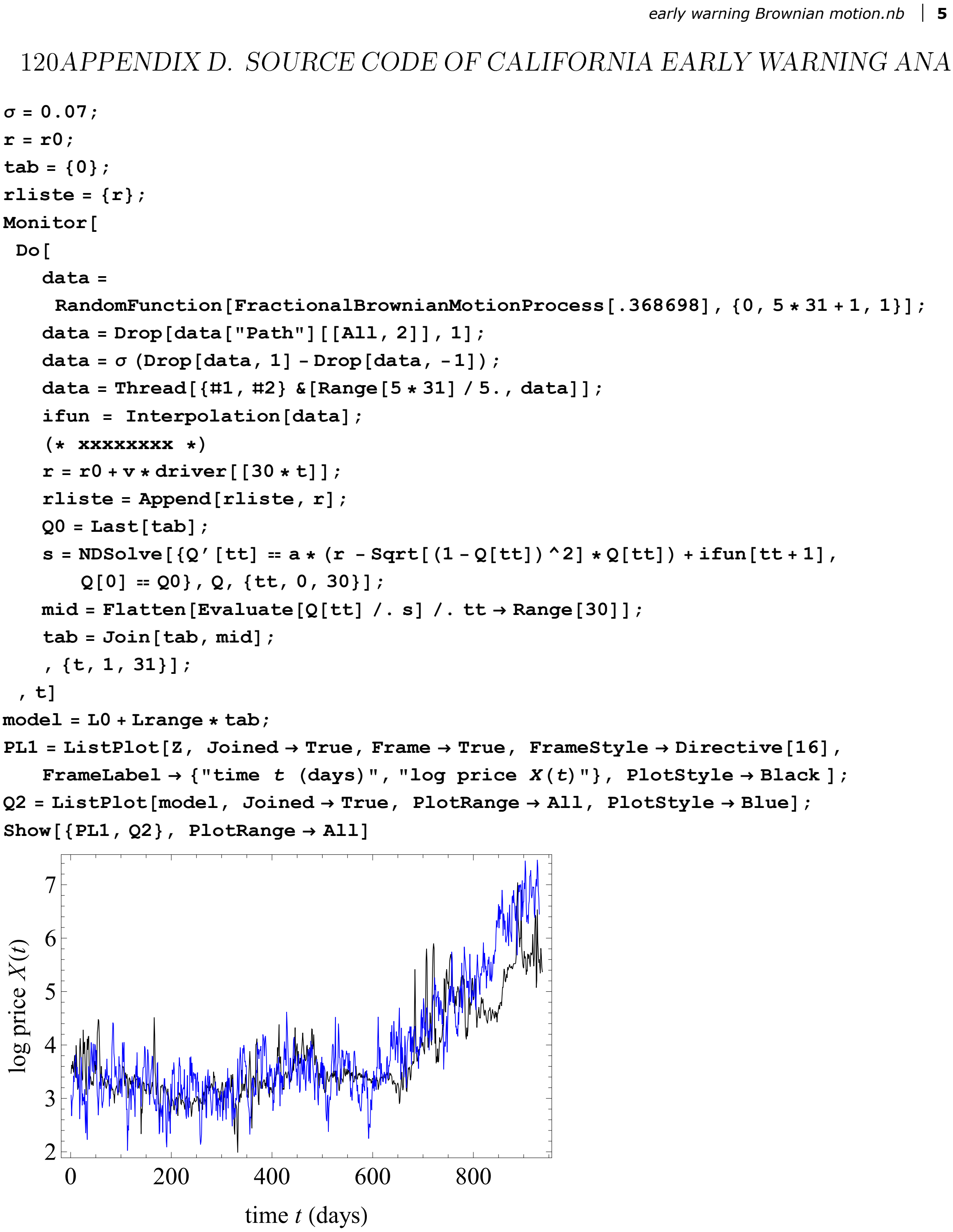


```
σ = 0.07;
r = r0;
tab = {0};
rliste = {r};
Monitor[
 Do[
   data =
    RandomFunction[FractionalBrownianMotionProcess[0.45], {0, 5 * 31 + 1, 1}];
   data = Drop[data["Path"][[All, 2]], 1];
   data = σ (Drop[data, 1] - Drop[data, -1]);
   data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
   ifun = Interpolation[data];
   (* xxxxxxxx *)
   r = r0 + v * driver[[30 * t]];
   rliste = Append[rliste, r];
   Q0 = Last[tab];
   s = NDSolve[{Q'[tt] == a * (r - Sqrt[(1 - Q[tt])^2] * Q[tt]) + ifun[tt + 1],
      Q[0] == Q0}, Q, {tt, 0, 30}];
   mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
   tab = Join[tab, mid];
   , {t, 1, 47}];
 , t]
pricemodel = tab;
```

```
PL1 = ListPlot[Drop[Drop[X, win / 2], 0],
    Joined → True, Frame → True, FrameStyle → Directive[16],
    FrameLabel → {"time t (days)", "log price X(t)"}, PlotStyle → Black ];
Q2 = ListPlot[L0 + Lrange * pricemodel, Joined → True,
    PlotRange → All, PlotStyle → Blue];
Show[{PL1, Q2}, PlotRange → All]
```

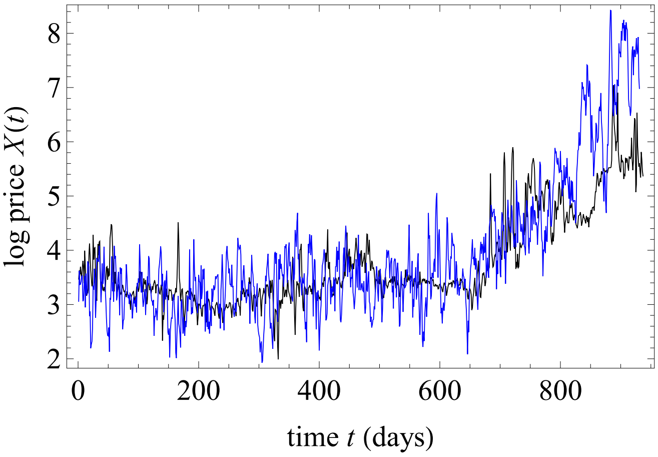


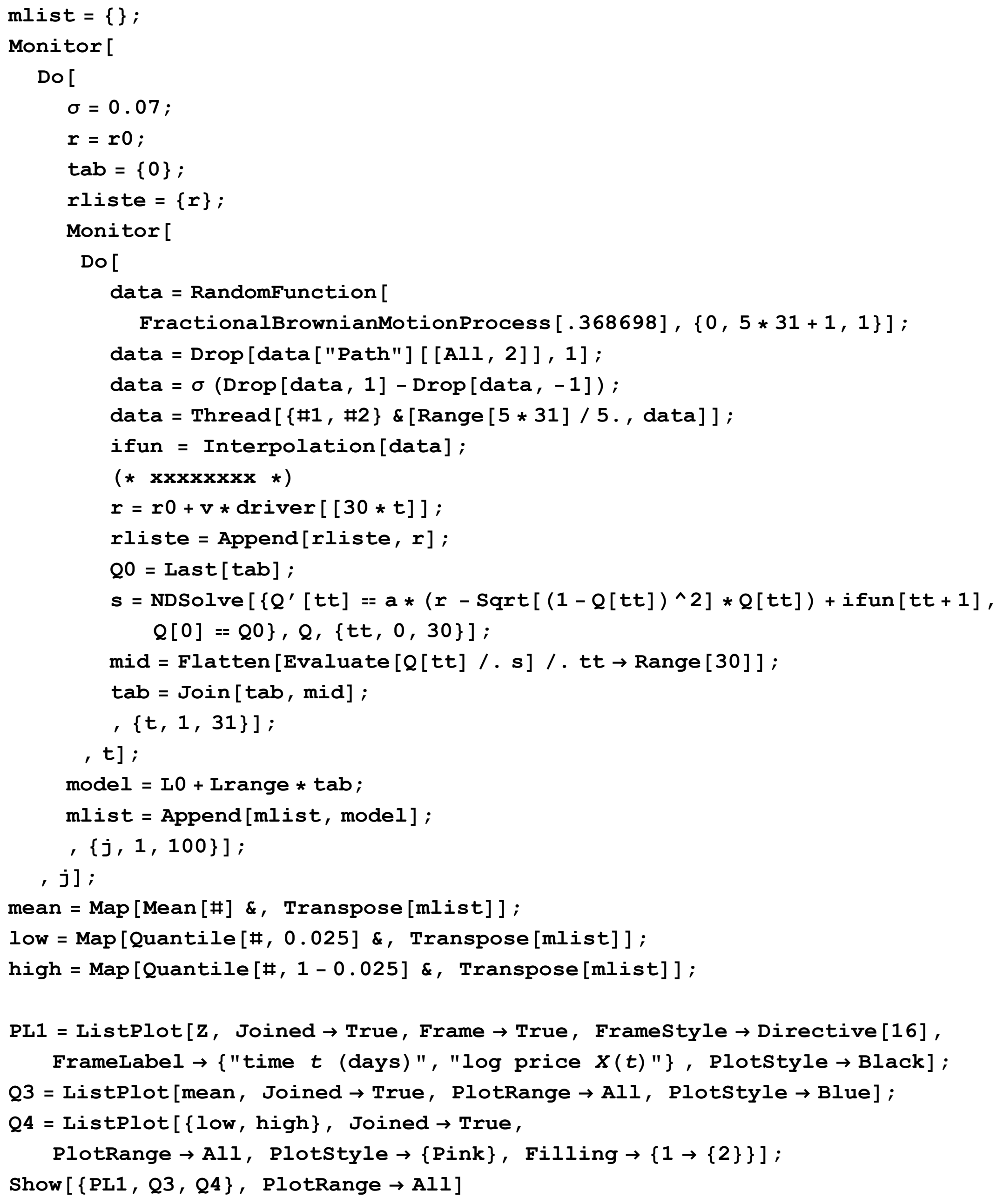
```
Z = Drop[Drop[X, win / 2], 0];
```



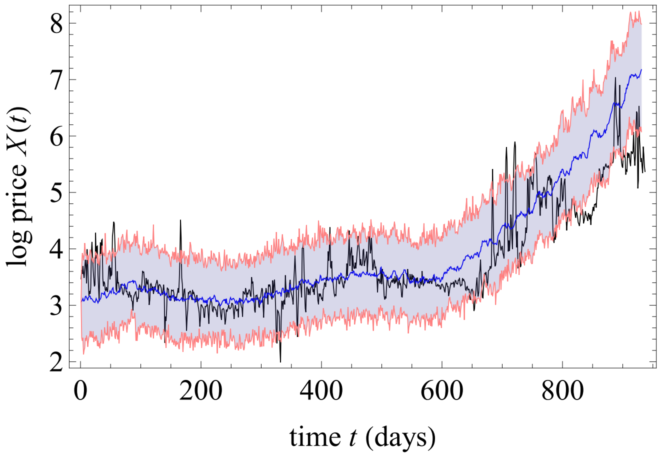
```
σ = 0.07;
r = r0;
tab = {0};
rliste = {r};
Monitor[
 Do[
   data =
    RandomFunction[FractionalBrownianMotionProcess[.368698], {0, 5 * 31 + 1, 1}];
   data = Drop[data["Path"][[All, 2]], 1];
   data = σ (Drop[data, 1] - Drop[data, -1]);
   data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
   ifun = Interpolation[data];
   (* xxxxxxxx *)
   r = r0 + v * driver[[30 * t]];
   rliste = Append[rliste, r];
   Q0 = Last[tab];
   s = NDSolve[{Q'[tt] == a * (r - Sqrt[(1 - Q[tt])^2] * Q[tt]) + ifun[tt + 1],
      Q[0] == Q0}, Q, {tt, 0, 30}];
   mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
   tab = Join[tab, mid];
   , {t, 1, 31}];
 , t]
model = L0 + Lrange * tab;
PL1 = ListPlot[Z, Joined → True, Frame → True, FrameStyle → Directive[16],
   FrameLabel → {"time  t  (days)", "log price  X(t)"}, PlotStyle → Black ];
Q2 = ListPlot[model, Joined → True, PlotRange → All, PlotStyle → Blue];
Show[{PL1, Q2}, PlotRange → All]
```

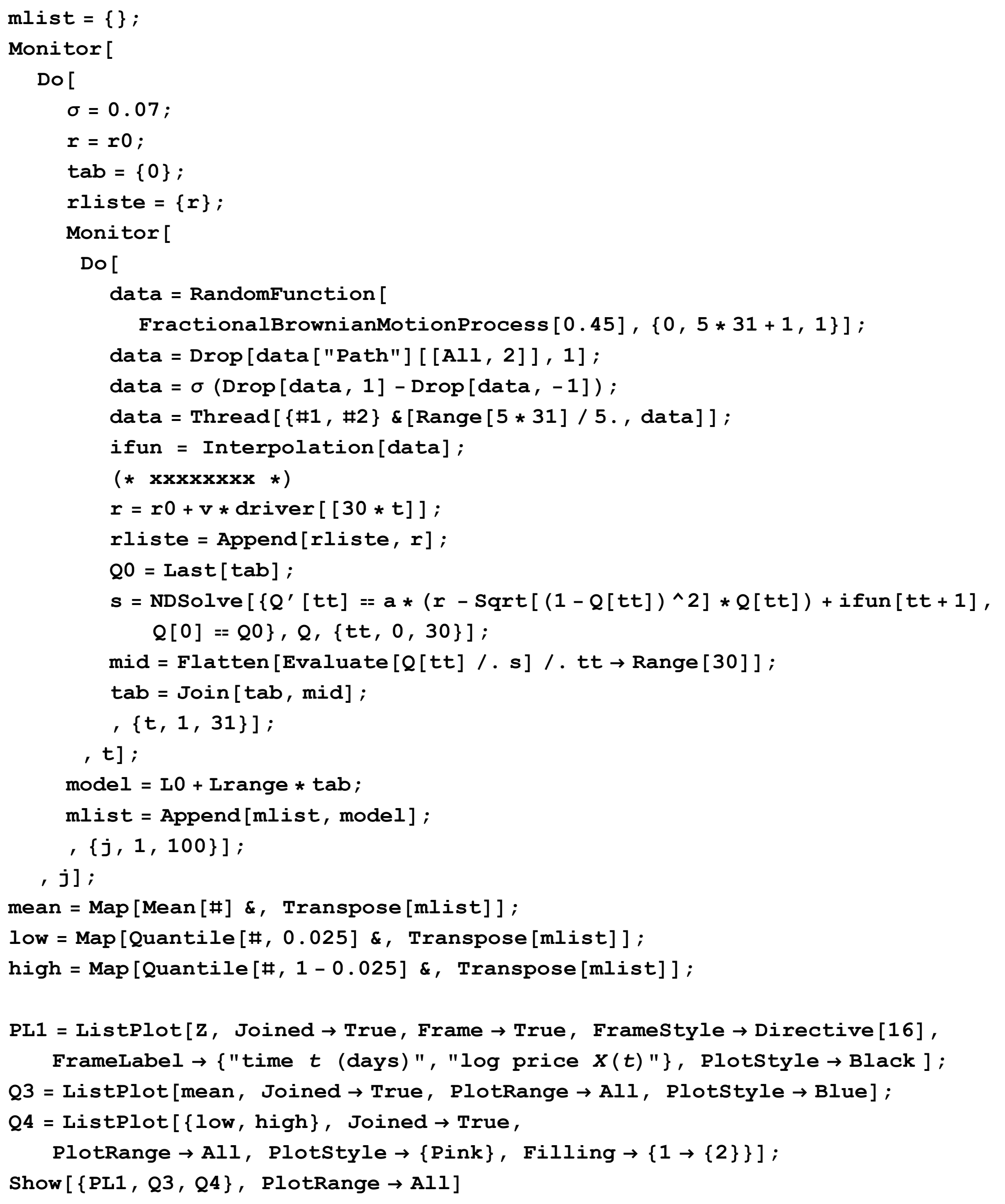
```
σ = 0.07;
r = r0;
tab = {0};
rliste = {r};
Monitor[
 Do[
   data =
    RandomFunction[FractionalBrownianMotionProcess[0.45], {0, 5 * 31 + 1, 1}];
   data = Drop[data["Path"][[All, 2]], 1];
   data = σ (Drop[data, 1] - Drop[data, -1]);
   data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
   ifun = Interpolation[data];
   (* xxxxxxxx *)
   r = r0 + v * driver[[30 * t]];
   rliste = Append[rliste, r];
   Q0 = Last[tab];
   s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt])^2] * Q[tt]) + ifun[tt + 1],
      Q[0] == Q0}, Q, {tt, 0, 30}];
   mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
   tab = Join[tab, mid];
   , {t, 1, 31}];
 , t]
model = L0 + Lrange * tab;
PL1 = ListPlot[Z, Joined → True, Frame → True, FrameStyle → Directive[16],
   FrameLabel → {"time  t  (days)", "log price  X(t)"}, PlotStyle → Black ];
Q2 = ListPlot[model, Joined → True, PlotRange → All, PlotStyle → Blue];
Show[{PL1, Q2}, PlotRange → All]
```



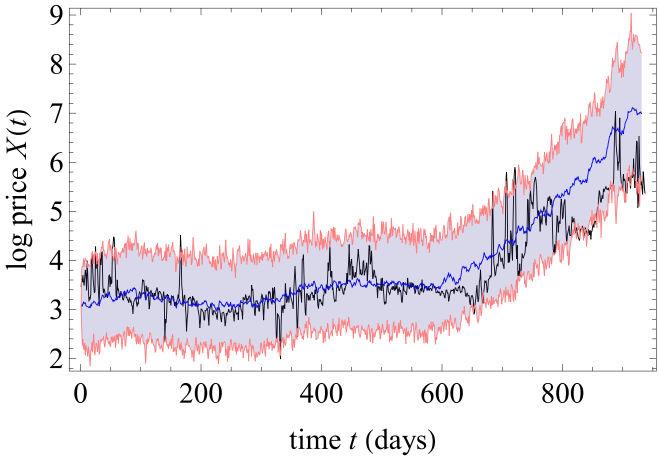


```
mlist = {};
Monitor[
  Do[
    σ = 0.07;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       data = RandomFunction[
         FractionalBrownianMotionProcess[.368698], {0, 5 * 31 + 1, 1}];
       data = Drop[data["Path"][[All, 2]], 1];
       data = σ (Drop[data, 1] - Drop[data, -1]);
       data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
       ifun = Interpolation[data];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] ⩵ a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
          Q[0] ⩵ Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 31}];
     , t];
    model = L0 + Lrange * tab;
    mlist = Append[mlist, model];
    , {j, 1, 100}];
  , j];
mean = Map[Mean[#] &, Transpose[mlist]];
low = Map[Quantile[#, 0.025] &, Transpose[mlist]];
high = Map[Quantile[#, 1 - 0.025] &, Transpose[mlist]];

PL1 = ListPlot[Z, Joined → True, Frame → True, FrameStyle → Directive[16],
   FrameLabel → {"time t (days)", "log price X(t)"} , PlotStyle → Black];
Q3 = ListPlot[mean, Joined → True, PlotRange → All, PlotStyle → Blue];
Q4 = ListPlot[{low, high}, Joined → True,
   PlotRange → All, PlotStyle → {Pink}, Filling → {1 → {2}}];
Show[{PL1, Q3, Q4}, PlotRange → All]
```


log price $X(t)$
2
3
4
5
6
7
8
0
200
400
600
800
time t (days)



```
mlist = {};
Monitor[
  Do[
    σ = 0.07;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       data = RandomFunction[
         FractionalBrownianMotionProcess[0.45], {0, 5 * 31 + 1, 1}];
       data = Drop[data["Path"][[All, 2]], 1];
       data = σ (Drop[data, 1] - Drop[data, -1]);
       data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
       ifun = Interpolation[data];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
          Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 31}];
     , t];
    model = L0 + Lrange * tab;
    mlist = Append[mlist, model];
    , {j, 1, 100}];
  , j];
mean = Map[Mean[#] &, Transpose[mlist]];
low = Map[Quantile[#, 0.025] &, Transpose[mlist]];
high = Map[Quantile[#, 1 - 0.025] &, Transpose[mlist]];

PL1 = ListPlot[Z, Joined → True, Frame → True, FrameStyle → Directive[16],
   FrameLabel → {"time t (days)", "log price X(t)"}, PlotStyle → Black ];
Q3 = ListPlot[mean, Joined → True, PlotRange → All, PlotStyle → Blue];
Q4 = ListPlot[{low, high}, Joined → True,
   PlotRange → All, PlotStyle → {Pink}, Filling → {1 → {2}}];
Show[{PL1, Q3, Q4}, PlotRange → All]
```



Early warning

```
σ = 0.07;
r = r0;
tab = {0};
rliste = {r};
Monitor[
  Do[
    data = RandomFunction[
      FractionalBrownianMotionProcess[.368698], {0, 5 * 31 + 1, 1}];
    data = Drop[data["Path"][[All, 2]], 1];
    data = σ (Drop[data, 1] - Drop[data, -1]);
    data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
    ifun = Interpolation[data];
    (* xxxxxxxx *)
    r = r0 + v * driver[[30 * t]];
    rliste = Append[rliste, r];
    Q0 = Last[tab];
    s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
        Q[0] == Q0}, Q, {tt, 0, 30}];
    mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
    tab = Join[tab, mid];
    , {t, 1, 31}];
  , t];
model = L0 + Lrange * tab;
```

```
r
```

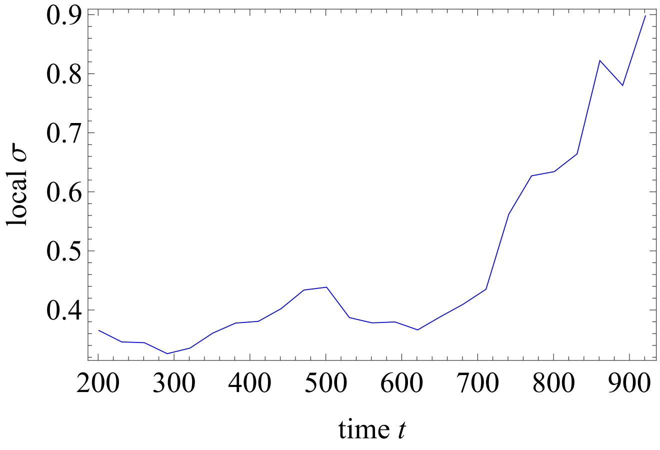
```
0.2052
```

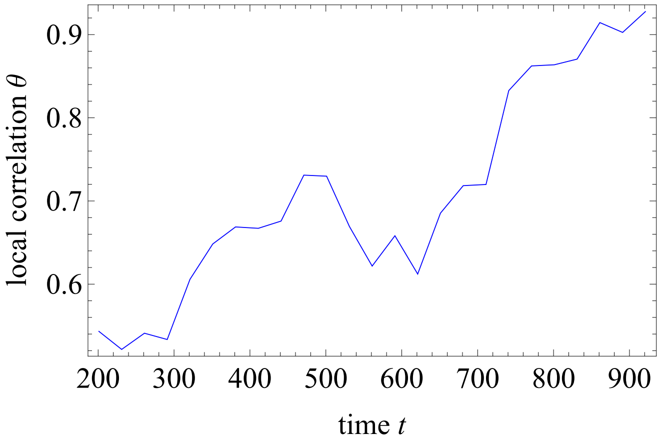
Since r<0.25 this is before the tipping point

```
θ = EstimatedProcess[model, OrnsteinUhlenbeckProcess[μ, σ, θ]][[3]]
```

```
0.202087
```

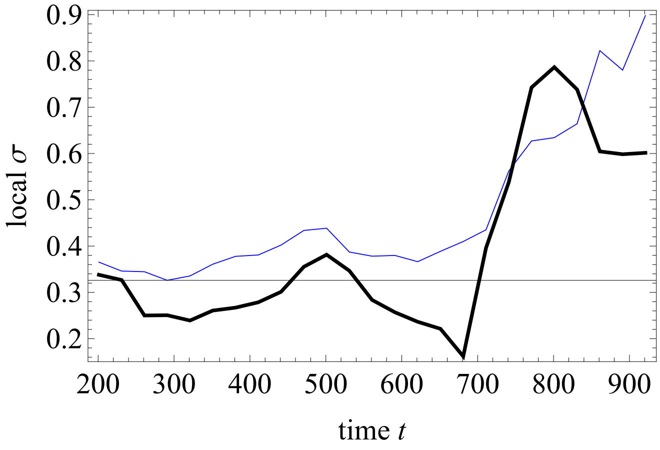
```
win = 200;
σlist = {};
τlist = {};
Monitor[
  Do[
    qqq = model[[t - win ;; t]];
    σloc = StandardDeviation[qqq]; (*local standard deviation*)
    τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
    (*local correlation scale*)
    σlist = Append[σlist, {t, σloc}];
    τlist = Append[τlist, {t, τloc}];
    , {t, win + 1, Length[model], 30}];
  , t];
QQ3 = ListPlot[σlist, Frame → True, FrameStyle → Directive[16],
   Joined → True, FrameLabel → {"time t", "local σ"}, PlotStyle → Blue];
QQ4 = ListPlot[τlist, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local correlation θ"}, PlotStyle → Blue];
Show[QQ3, PlotRange → All]
Show[QQ4, PlotRange → All]
```

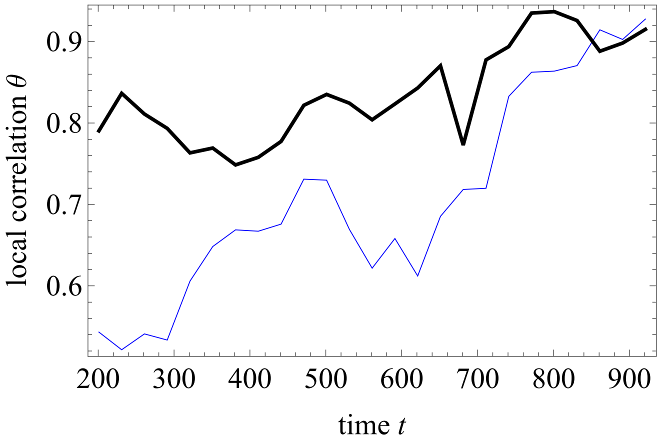


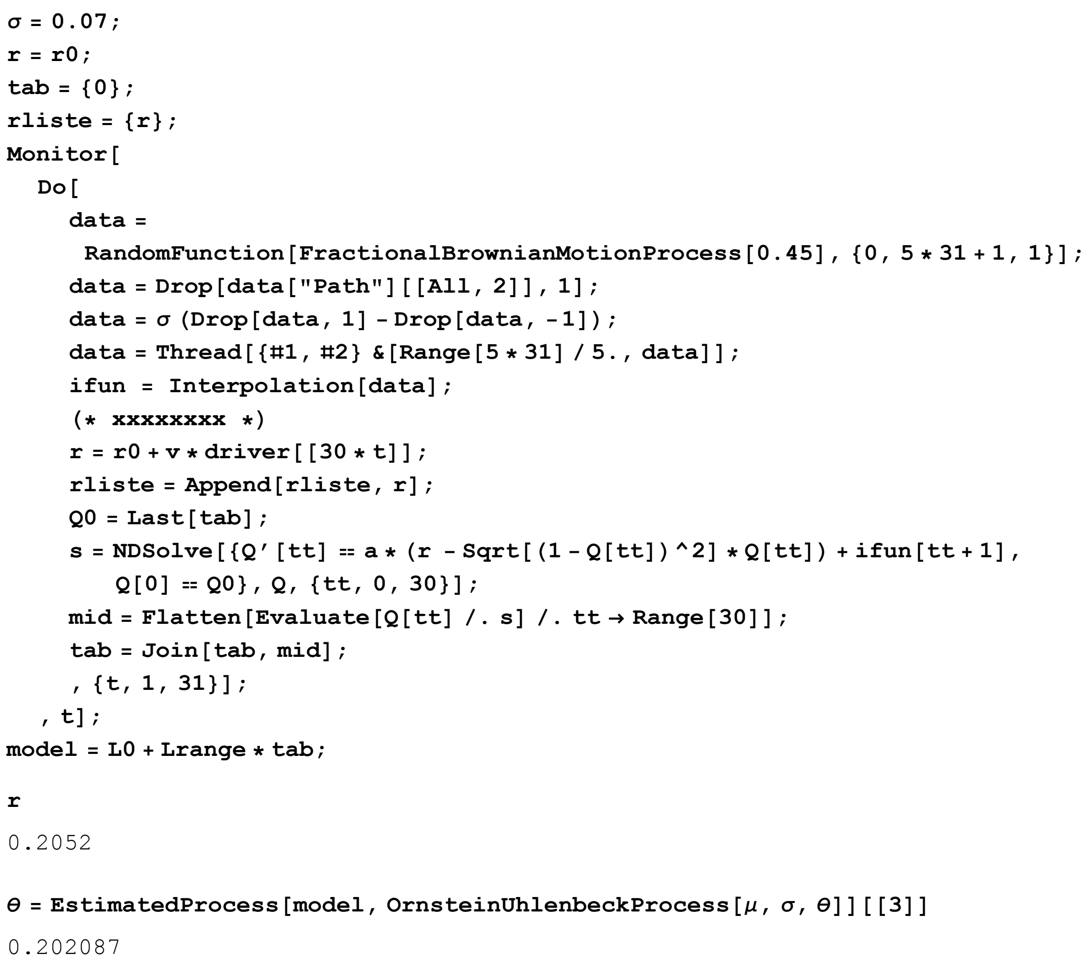


real data:

```
win = 200;
σlist = {};
τlist = {};
Monitor[
  Do[
    qqq = Z[[t - win ;; t]];
    σloc = StandardDeviation[qqq]; (*local standard deviation*)
    τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
    (*local correlation scale*)
    σlist = Append[σlist, {t, σloc}];
    τlist = Append[τlist, {t, τloc}];
    , {t, win + 1, Length[Z], 30}];
  , t];
QQ5 = ListPlot[σlist, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local σ"}, PlotStyle → {Thick, Black}];
QQ6 = ListPlot[τlist, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local correlation θ"}, PlotStyle → {Thick, Black}];
Show[{QQ3, QQ5}, PlotRange → All]
Show[{QQ4, QQ6}, PlotRange → All]
```







```
σ = 0.07;
r = r0;
tab = {0};
rliste = {r};
Monitor[
  Do[
    data =
     RandomFunction[FractionalBrownianMotionProcess[0.45], {0, 5 * 31 + 1, 1}];
    data = Drop[data["Path"][[All, 2]], 1];
    data = σ (Drop[data, 1] - Drop[data, -1]);
    data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
    ifun = Interpolation[data];
    (* xxxxxxxx *)
    r = r0 + v * driver[[30 * t]];
    rliste = Append[rliste, r];
    Q0 = Last[tab];
    s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
       Q[0] == Q0}, Q, {tt, 0, 30}];
    mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
    tab = Join[tab, mid];
    , {t, 1, 31}];
  , t];
model = L0 + Lrange * tab;
```

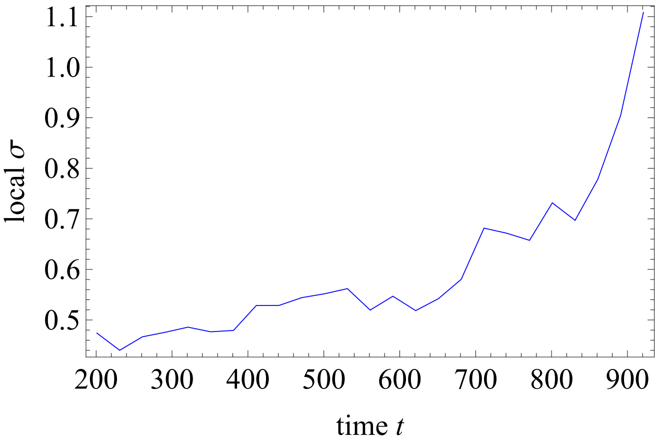
```
r
```

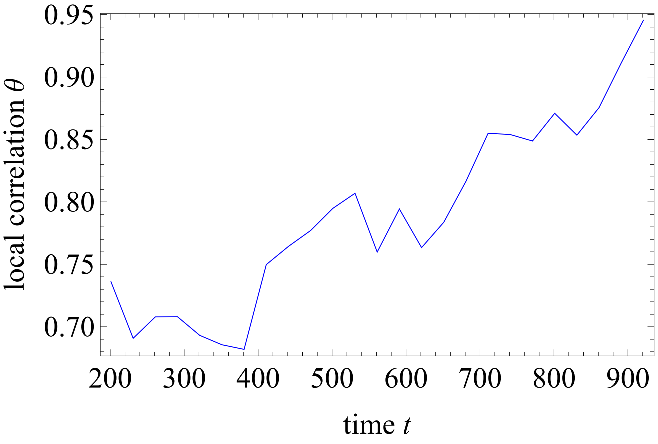
```
0.2052
```

```
θ = EstimatedProcess[model, OrnsteinUhlenbeckProcess[μ, σ, θ]][[3]]
```

```
0.202087
```

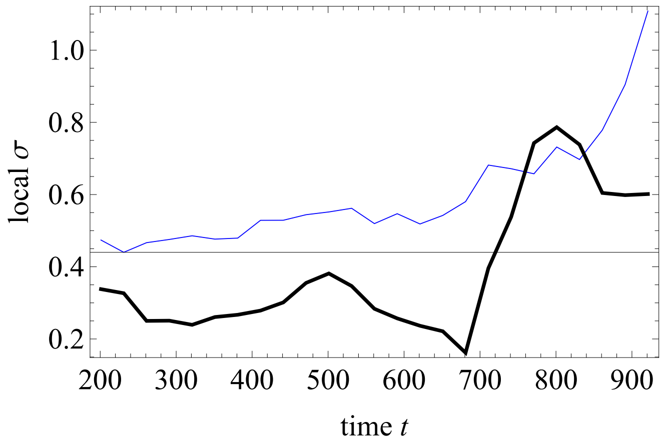
```
win = 200;
σlist = {};
τlist = {};
Monitor[
  Do[
    qqq = model[[t - win ;; t]];
    σloc = StandardDeviation[qqq]; (*local standard deviation*)
    τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
    (*local correlation scale*)
    σlist = Append[σlist, {t, σloc}];
    τlist = Append[τlist, {t, τloc}];
    , {t, win + 1, Length[model], 30}];
  , t];
QQ3 = ListPlot[σlist, Frame → True, FrameStyle → Directive[16],
   Joined → True, FrameLabel → {"time t", "local σ"}, PlotStyle → Blue];
QQ4 = ListPlot[τlist, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local correlation θ"}, PlotStyle → Blue];
Show[QQ3, PlotRange → All]
Show[QQ4, PlotRange → All]
```

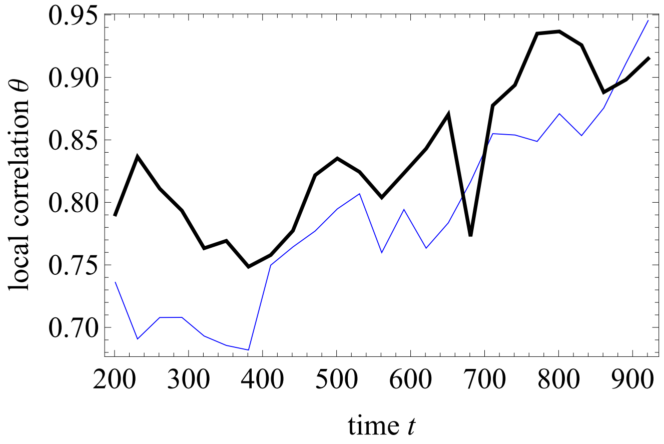




real data

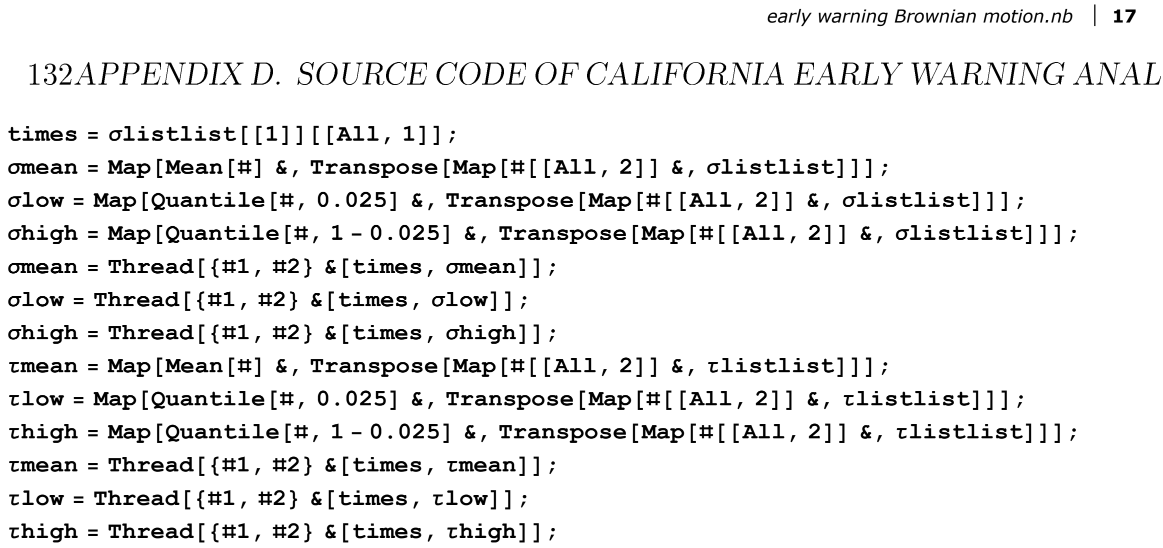
```
win = 200;
σlist = {};
τlist = {};
Monitor[
  Do[
    qqq = Z[[t - win ;; t]];
    σloc = StandardDeviation[qqq]; (*local standard deviation*)
    τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
    (*local correlation scale*)
    σlist = Append[σlist, {t, σloc}];
    τlist = Append[τlist, {t, τloc}];
    , {t, win + 1, Length[Z], 30}];
  , t];
QQ5 = ListPlot[σlist, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local σ"}, PlotStyle → {Thick, Black}];
QQ6 = ListPlot[τlist, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local correlation θ"}, PlotStyle → {Thick, Black}];
Show[{QQ3, QQ5}, PlotRange → All]
Show[{QQ4, QQ6}, PlotRange → All]
```





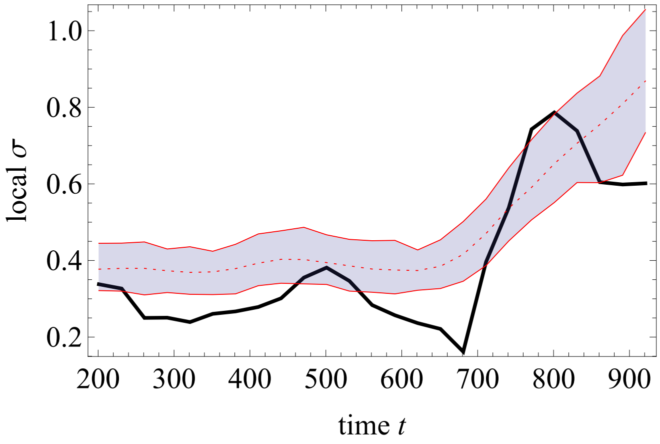
monte carlo

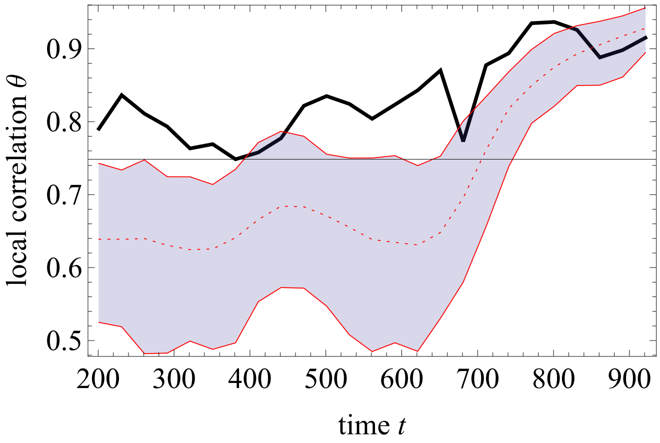
```
τlistlist = {};
σlistlist = {};
Monitor[
  Do[
    σ = 0.07;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       data = RandomFunction[
         FractionalBrownianMotionProcess[.368698], {0, 5 * 31 + 1, 1}];
       data = Drop[data["Path"][[All, 2]], 1];
       data = σ (Drop[data, 1] - Drop[data, -1]);
       data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
       ifun = Interpolation[data];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q'[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
          Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 31}];
     , t];
    model = L0 + Lrange * tab;
    win = 200;
    σlist = {};
    τlist = {};
    Monitor[
     Do[
       qqq = model[[t - win ;; t]];
       σloc = StandardDeviation[qqq]; (*local standard deviation*)
       τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
       (*local correlation scale*)
       σlist = Append[σlist, {t, σloc}];
       τlist = Append[τlist, {t, τloc}];
       , {t, win + 1, Length[model], 30}];
     , t];
    τlistlist = Append[τlistlist, τlist];
    σlistlist = Append[σlistlist, σlist];
    , {u, 1, 100}];
  , u];
```



```
times = σlistlist[[1]][[All, 1]];
σmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σmean = Thread[{#1, #2} &[times, σmean]];
σlow = Thread[{#1, #2} &[times, σlow]];
σhigh = Thread[{#1, #2} &[times, σhigh]];
τmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τmean = Thread[{#1, #2} &[times, τmean]];
τlow = Thread[{#1, #2} &[times, τlow]];
τhigh = Thread[{#1, #2} &[times, τhigh]];
```

```
QQQ1 = ListPlot[{σlow, σhigh},
    Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local σ"}, PlotStyle → Red, Filling → {1 → {2}}];
QQQ2 = ListPlot[σmean, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local σ"}, PlotStyle → {Red, Dotted}];
QQQ3 = ListPlot[{τlow, τhigh}, Frame → True, FrameStyle → Directive[16],
    Joined → True, FrameLabel → {"time t", "local σ"},
    PlotStyle → Red, Filling → {1 → {2}}];
QQQ4 = ListPlot[τmean, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local σ"}, PlotStyle → {Red, Dotted}];
Show[{QQ5, QQQ1, QQQ2}, PlotRange → All]
Show[{QQ6, QQQ3, QQQ4}, PlotRange → All]
```





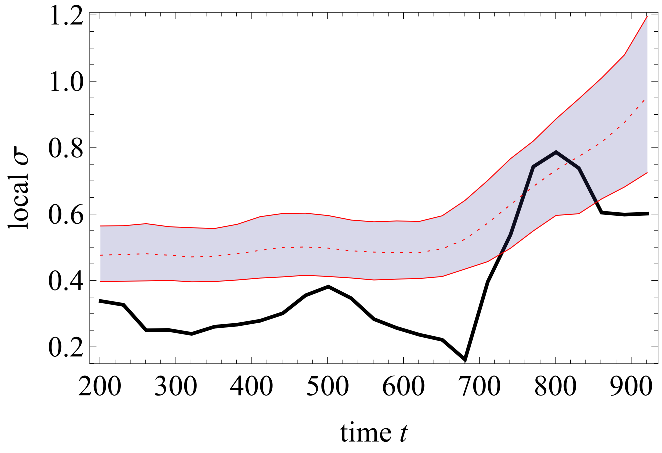
```
Length[Z] / 30.
```

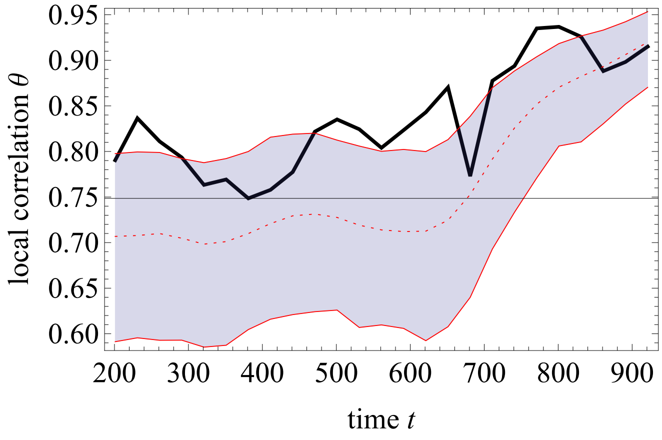
```
31.2333
```

```
τlistlist = {};
σlistlist = {};
Monitor[
  Do[
    σ = 0.07;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       data = RandomFunction[
         FractionalBrownianMotionProcess[0.45], {0, 5 * 31 + 1, 1}];
       data = Drop[data["Path"][[All, 2]], 1];
       data = σ (Drop[data, 1] - Drop[data, -1]);
       data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
       ifun = Interpolation[data];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt])^2] * Q[tt]) + ifun[tt + 1],
          Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 31}];
     , t];
    model = L0 + Lrange * tab;
    win = 200;
    σlist = {};
    τlist = {};
    Monitor[
     Do[
       qqq = model[[t - win ;; t]];
       σloc = StandardDeviation[qqq]; (*local standard deviation*)
       τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
       (*local correlation scale*)
       σlist = Append[σlist, {t, σloc}];
       τlist = Append[τlist, {t, τloc}];
       , {t, win + 1, Length[model], 30}];
     , t];
    τlistlist = Append[τlistlist, τlist];
    σlistlist = Append[σlistlist, σlist];
    , {u, 1, 1000}];
  , u];
```

```
times = σlistlist[[1]][[All, 1]];
σmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σmean = Thread[{#1, #2} &[times, σmean]];
σlow = Thread[{#1, #2} &[times, σlow]];
σhigh = Thread[{#1, #2} &[times, σhigh]];
τmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τmean = Thread[{#1, #2} &[times, τmean]];
τlow = Thread[{#1, #2} &[times, τlow]];
τhigh = Thread[{#1, #2} &[times, τhigh]];
```

```
QQQ1 = ListPlot[{σlow, σhigh},
   Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local σ"}, PlotStyle → Red, Filling → {1 → {2}}];
QQQ2 = ListPlot[σmean, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local σ"}, PlotStyle → {Red, Dotted}];
QQQ3 = ListPlot[{τlow, τhigh}, Frame → True, FrameStyle → Directive[16],
   Joined → True, FrameLabel → {"time t", "local σ"},
   PlotStyle → Red, Filling → {1 → {2}}];
QQQ4 = ListPlot[τmean, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local σ"}, PlotStyle → {Red, Dotted}];
Show[{QQ5, QQQ1, QQQ2}, PlotRange → All]
Show[{QQ6, QQQ3, QQQ4}, PlotRange → All]
```





Local Hurst exponent

A function for de-trended fluctuation analysis:

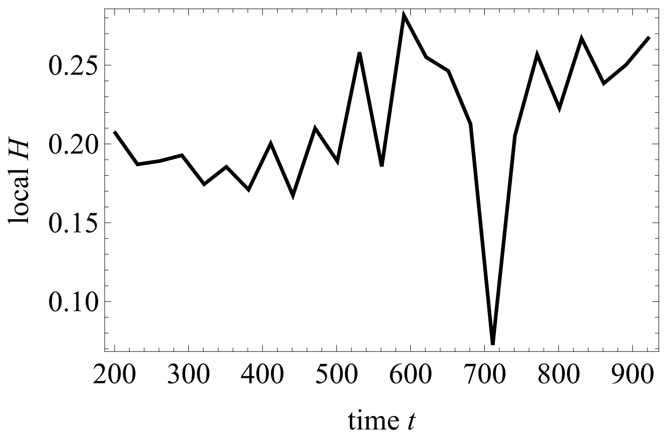
```
fit[z_, o_] := Fit[z, zz^Range[0, o], zz] /. zz → Range[Length[z]];
DFA[x_, o_, imin_, imax_] := (Module[{L, i, X, Z, var, detrends}, var = {};
    Do[L = 2^i;
     X = FoldList[Plus, 0, x - Mean[x]];
     Z = Partition[X, L];
     detrends = Map[# - fit[#, o] &, Z];
     var = Append[var, {L, Sqrt[Mean[Flatten[detrends]^2]]}];,
     {i, imin, imax}];
    Return[var];];)
```

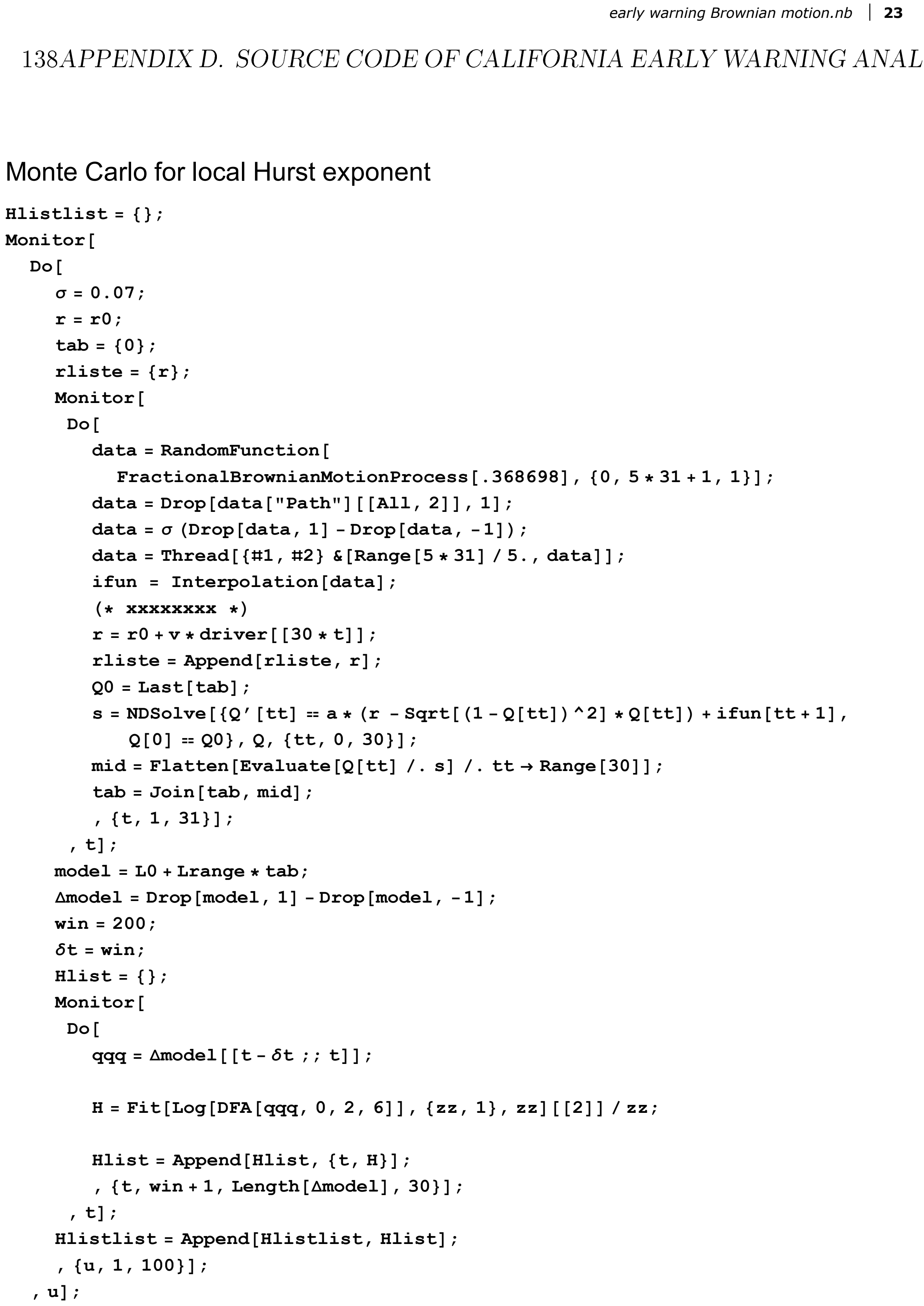
```
Δmodel = Drop[model, 1] - Drop[model, -1];
Fit[Log[DFA[Δmodel, 1, 2, 9]], {zz, 1}, zz]
```

```
-1.7131 + 0.185799 zz
```

```
δt = 200;
t = δt + 1;
Δmodel = Drop[model, 1] - Drop[model, -1];
H = Fit[Log[DFA[Δmodel[[t - δt ;; t]], 1, 2, 7]], {zz, 1}, zz][[2]] / zz;
```

```
win = 200;
δt = win;
Hlist = {};
Monitor[
  Do[
    ΔZ = Drop[Z, 1] - Drop[Z, -1];
    qqq = ΔZ[[t - δt ;; t]];

    H = Fit[Log[DFA[qqq, 0, 2, 6]], {zz, 1}, zz][[2]] / zz;

    Hlist = Append[Hlist, {t, H}];
    , {t, win + 1, Length[Z], 30}];
  , t];
QQ3 = ListPlot[Hlist, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local H"}, PlotStyle → {Black, Thick}];
Show[QQ3, PlotRange → All]
```



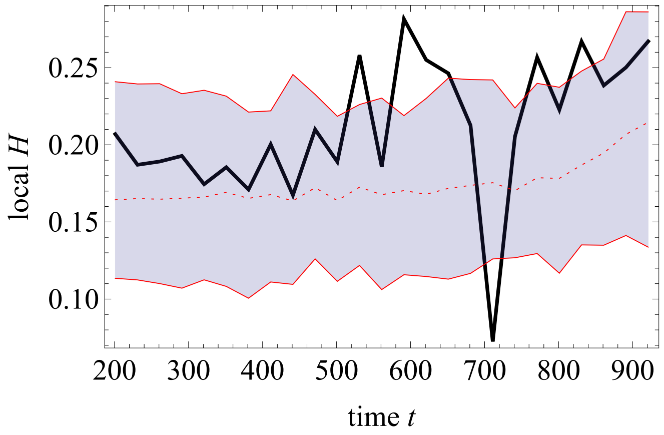


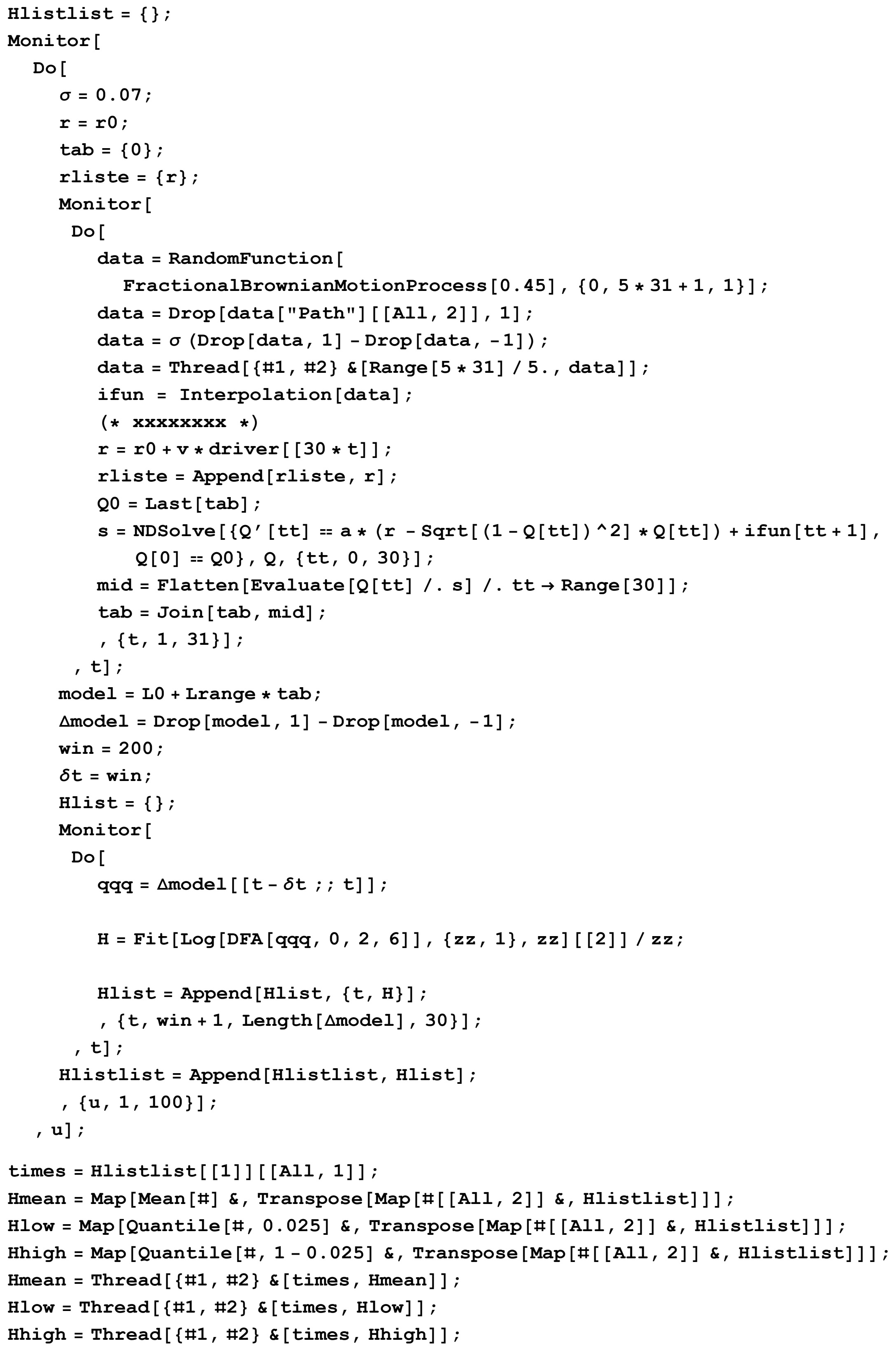
Monte Carlo for local Hurst exponent

```
Hlistlist = {};
Monitor[
  Do[
    σ = 0.07;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       data = RandomFunction[
         FractionalBrownianMotionProcess[.368698], {0, 5 * 31 + 1, 1}];
       data = Drop[data["Path"][[All, 2]], 1];
       data = σ (Drop[data, 1] - Drop[data, -1]);
       data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
       ifun = Interpolation[data];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
          Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 31}];
     , t];
    model = L0 + Lrange * tab;
    Δmodel = Drop[model, 1] - Drop[model, -1];
    win = 200;
    δt = win;
    Hlist = {};
    Monitor[
     Do[
       qqq = Δmodel[[t - δt ;; t]];

       H = Fit[Log[DFA[qqq, 0, 2, 6]], {zz, 1}, zz][[2]] / zz;

       Hlist = Append[Hlist, {t, H}];
       , {t, win + 1, Length[Δmodel], 30}];
     , t];
    Hlistlist = Append[Hlistlist, Hlist];
    , {u, 1, 100}];
  , u];
```

```
times = Hlistlist[[1]][[All, 1]];
Hmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hmean = Thread[{#1, #2} &[times, Hmean]];
Hlow = Thread[{#1, #2} &[times, Hlow]];
Hhigh = Thread[{#1, #2} &[times, Hhigh]];
```

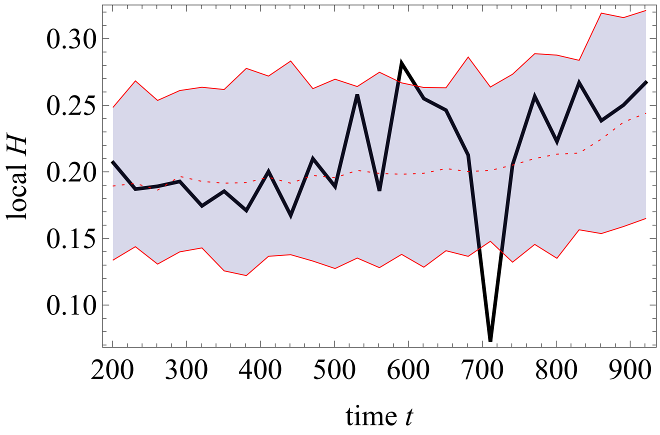
```
QQ5 = ListPlot[{Hlow, Hhigh}, Frame → True, FrameStyle → Directive[16],
    Joined → True, FrameLabel → {"time t", "local \!\(\*
StyleBox[\"H\",\nFontSlant->\"Italic\"]\)"},
    PlotStyle → Red, Filling → {1 → {2}}];
QQ7 = ListPlot[Hmean, Frame → True, FrameStyle → Directive[16],
    Joined → True, FrameLabel → {"time t", "local \!\(\*
StyleBox[\"H\",\nFontSlant->\"Italic\"]\)"}, PlotStyle → {Red, Dotted}];
Show[{QQ3, QQ5, QQ7}, PlotRange → All]
```





```
Hlistlist = {};
Monitor[
  Do[
    σ = 0.07;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       data = RandomFunction[
         FractionalBrownianMotionProcess[0.45], {0, 5 * 31 + 1, 1}];
       data = Drop[data["Path"][[All, 2]], 1];
       data = σ (Drop[data, 1] - Drop[data, -1]);
       data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
       ifun = Interpolation[data];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
          Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 31}];
     , t];
    model = L0 + Lrange * tab;
    Δmodel = Drop[model, 1] - Drop[model, -1];
    win = 200;
    δt = win;
    Hlist = {};
    Monitor[
     Do[
       qqq = Δmodel[[t - δt ;; t]];

       H = Fit[Log[DFA[qqq, 0, 2, 6]], {zz, 1}, zz][[2]] / zz;

       Hlist = Append[Hlist, {t, H}];
       , {t, win + 1, Length[Δmodel], 30}];
     , t];
    Hlistlist = Append[Hlistlist, Hlist];
    , {u, 1, 100}];
  , u];

times = Hlistlist[[1]][[All, 1]];
Hmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hmean = Thread[{#1, #2} &[times, Hmean]];
Hlow = Thread[{#1, #2} &[times, Hlow]];
Hhigh = Thread[{#1, #2} &[times, Hhigh]];
```

```
QQ5 = ListPlot[{Hlow, Hhigh}, Frame → True, FrameStyle → Directive[16],
    Joined → True, FrameLabel → {"time t", "local \!\(\*
StyleBox[\"H\",\nFontSlant->\"Italic\"]\)"},
    PlotStyle → Red, Filling → {1 → {2}}];
QQ7 = ListPlot[Hmean, Frame → True, FrameStyle → Directive[16],
    Joined → True, FrameLabel → {"time t", "local \!\(\*
StyleBox[\"H\",\nFontSlant->\"Italic\"]\)"}, PlotStyle → {Red, Dotted}];
Show[{QQ3, QQ5, QQ7}, PlotRange → All]
```

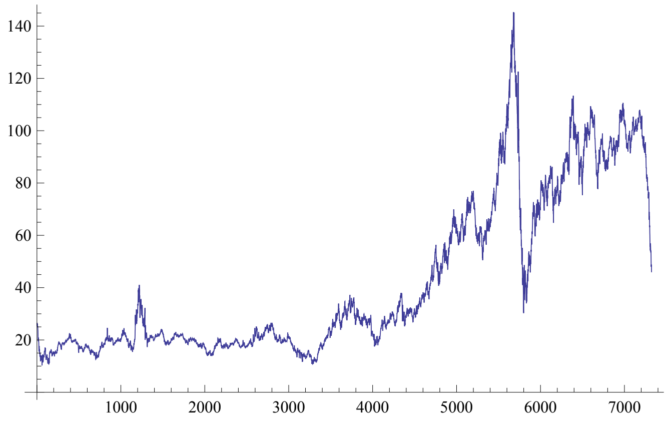


Appendix E

Source code of oil prices analysis

This appendix contains analysis of oil prices. The code plots average daily oil prices of 1986-2014 years. We use fractional Brownian motion process with $H = 0.3$. Then find the early-warning signals to predict the crisis.

```
X = ReadList["C:\\California electricity crisis/oilprice.txt"][[All, 2]];
ListPlot[X, Joined → True]
```



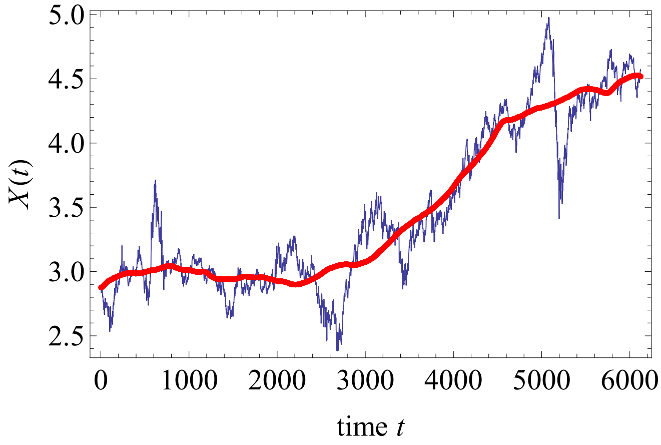
```
X = Flatten[X];
```

```
Length[X]
```

```
7323
```

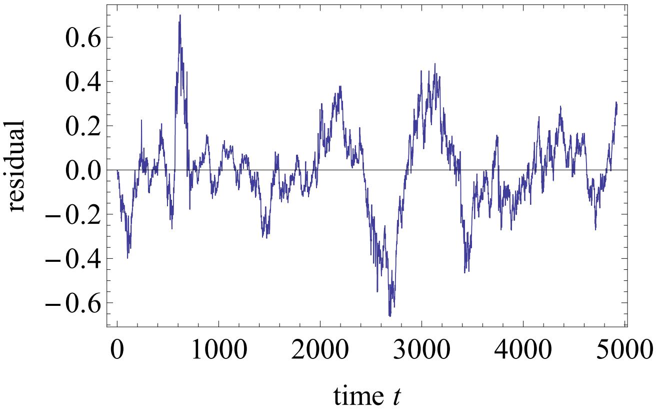
We have 7323 days

```
win = 1200;
X = Log[X];
L = MovingAverage[X, win];
PL1 = ListPlot[Drop[Drop[X, win / 2], -win / 2], Joined → True, Frame → True,
   FrameStyle → Directive[16], FrameLabel → {"time  t", "X(t)"} ];
PL2 = ListPlot[L, PlotStyle → Red];
residual = Drop[Drop[X, win / 2 - 1], -win / 2] - L;
PL0 = Show[PL1, PL2]
```



Residual before the crisis:

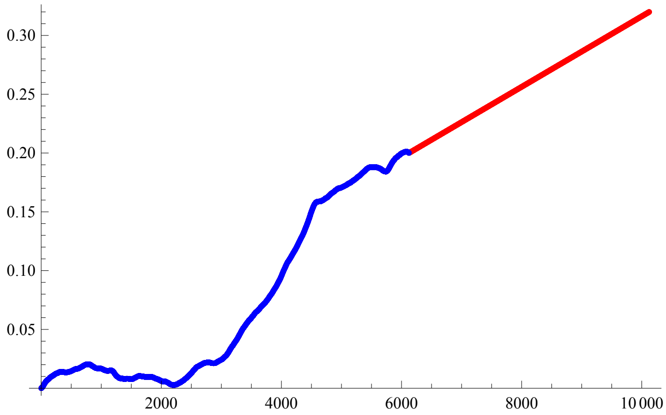
```
ListPlot[Drop[residual, -1200],
 Joined → True, Frame → True, FrameStyle → Directive[16],
 FrameLabel → {"time t", "residual"}, PlotRange → All ]
```



```
θ = EstimatedProcess[Drop[residual, -1000],
    OrnsteinUhlenbeckProcess[μ, σ, θ]][[3]];
Clear[a]
sol = Solve[a Sqrt[1 - 4 * 0.14] == θ, a];
a = sol[[1]][[1]][[2]]
```

```
0.0106574
```

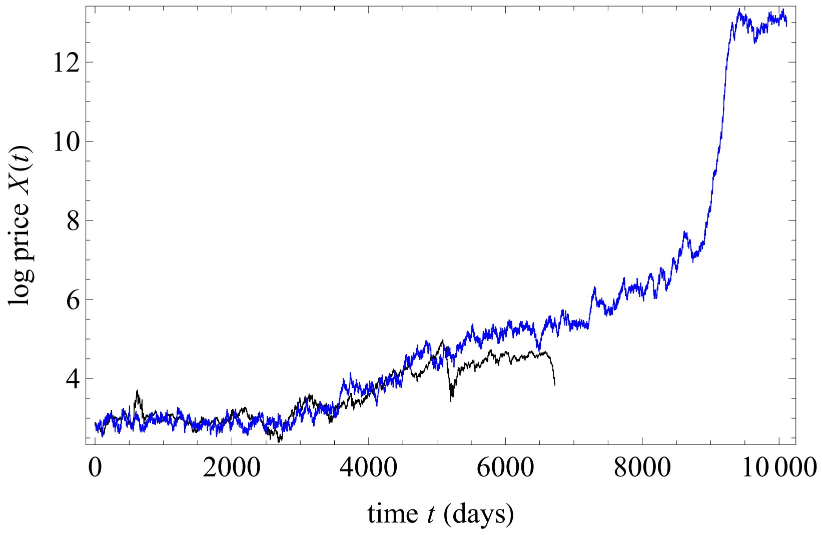
```
L0 = First[L];
Lrange = Last[L] - First[L];
Lrange = Lrange / 0.2;
driver = (L - L0) / Lrange;
r0 = 0.0;
v = 1;
PL0 = ListPlot[r0 + v * driver, PlotStyle → Blue];
driver = Join[driver, Last[driver] + 0.00003 * Range[4000]];
PL1 = ListPlot[r0 + v * driver, PlotStyle → Red];
Show[PL1, PL0]
```



```
Length[driver] / 30.
```

```
337.467
```

```
σ = 0.02;
r = r0;
tab = {0};
rliste = {r};
Monitor[
 Do[
   data =
    RandomFunction[FractionalBrownianMotionProcess[0.3], {0, 5 * 31 + 1, 1}];
   data = Drop[data["Path"][[All, 2]], 1];
   data = σ (Drop[data, 1] - Drop[data, -1]);
   data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
   ifun  = Interpolation[data];
   (* xxxxxxxx *)
   r = r0 + v * driver[[30 * t]];
   rliste = Append[rliste, r];
   Q0 = Last[tab];
   s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt])^2] * Q[tt]) + ifun[tt + 1],
      Q[0] == Q0}, Q, {tt, 0, 30}];
   mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
   tab = Join[tab, mid];
   , {t, 1, 337}];
 , t]
pricemodel = tab;
PL1 = ListPlot[Drop[Drop[X, win / 2], 0],
   Joined → True, Frame → True,  FrameStyle → Directive[16],
   FrameLabel → {"time  t  (days)", "log price  X(t)"} , PlotStyle → Black];
Q2 = ListPlot[L0 + Lrange * pricemodel,  Joined → True,
   PlotRange → All,  PlotStyle → Blue];
Show[{PL1, Q2},  PlotRange → All]
```

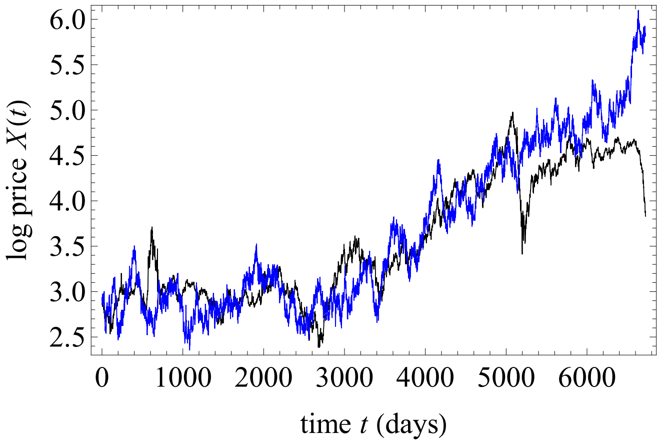


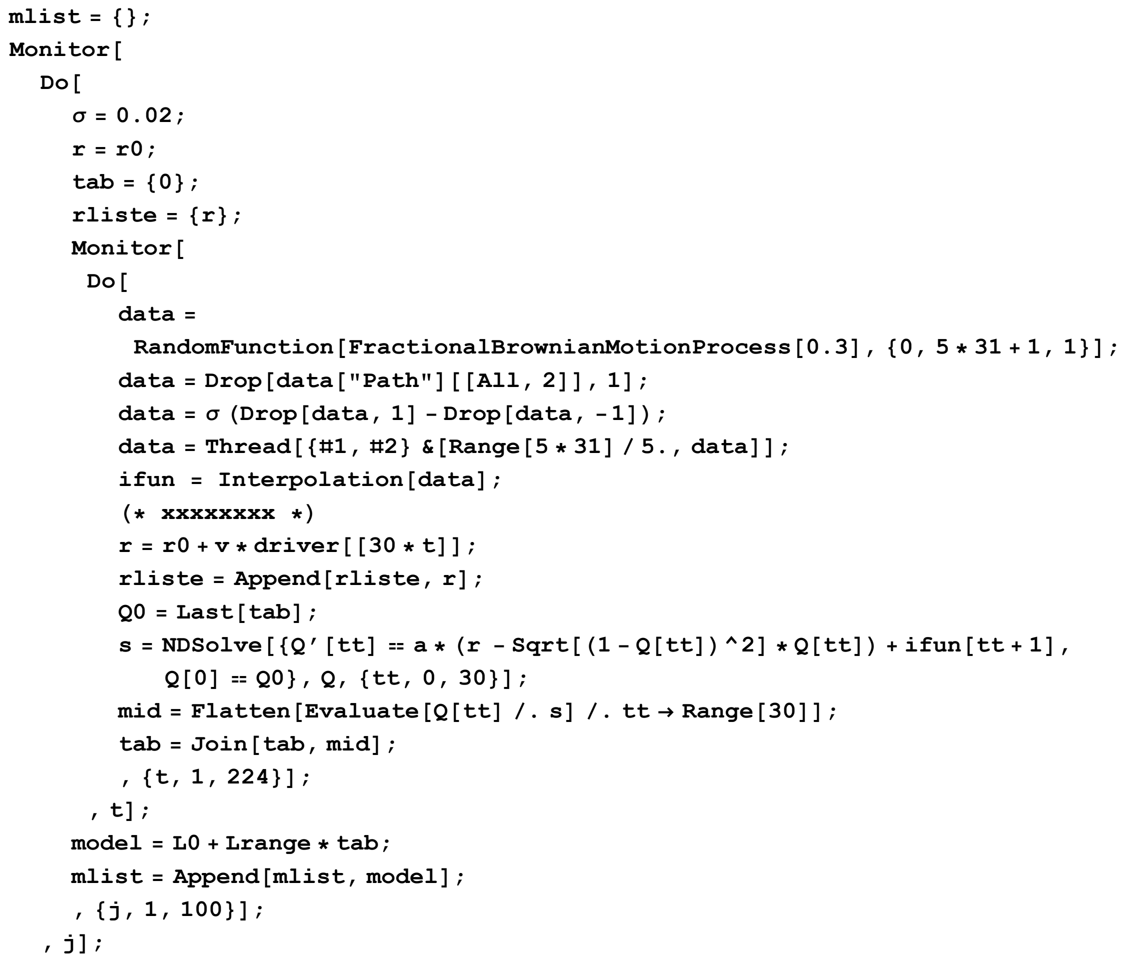
```
Z = Drop[Drop[X, win / 2], 0];
```

```
Length[Z] / 30.
```

```
224.1
```

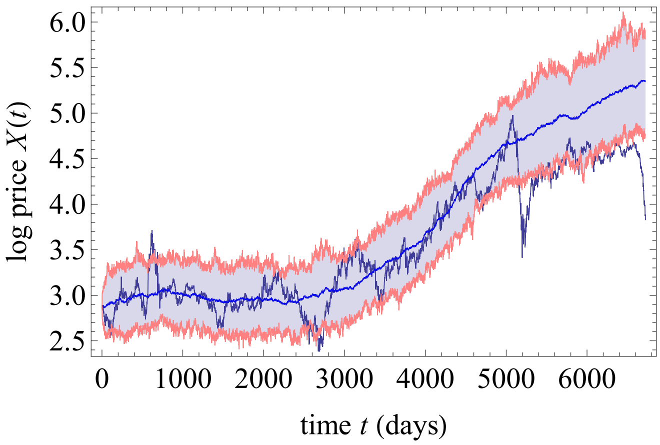
```
σ = 0.02;
r = r0;
tab = {0};
rliste = {r};
Monitor[
 Do[
   data =
    RandomFunction[FractionalBrownianMotionProcess[0.3], {0, 5 * 31 + 1, 1}];
   data = Drop[data["Path"][[All, 2]], 1];
   data = σ (Drop[data, 1] - Drop[data, -1]);
   data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
   ifun  = Interpolation[data];
   (* xxxxxxxx *)
   r = r0 + v * driver[[30 * t]];
   rliste = Append[rliste, r];
   Q0 = Last[tab];
   s = NDSolve[{Q′[tt] ⩵ a * (r - Sqrt[(1 - Q[tt])^2] * Q[tt]) + ifun[tt + 1],
      Q[0] ⩵ Q0}, Q, {tt, 0, 30}];
   mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
   tab = Join[tab, mid];
   , {t, 1, 224}];
 , t]
model = L0 + Lrange * tab;
PL1 = ListPlot[Z, Joined → True, Frame → True, FrameStyle → Directive[16],
   FrameLabel → {"time  t  (days)", "log price  X(t)"}, PlotStyle → Black ];
Q2 = ListPlot[model, Joined → True, PlotRange → All, PlotStyle → Blue];
Show[{PL1, Q2}, PlotRange → All]
```





```
mlist = {};
Monitor[
  Do[
    σ = 0.02;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       data =
        RandomFunction[FractionalBrownianMotionProcess[0.3], {0, 5 * 31 + 1, 1}];
       data = Drop[data["Path"][[All, 2]], 1];
       data = σ (Drop[data, 1] - Drop[data, -1]);
       data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
       ifun = Interpolation[data];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
          Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 224}];
     , t];
    model = L0 + Lrange * tab;
    mlist = Append[mlist, model];
    , {j, 1, 100}];
  , j];
```

```
mean = Map[Mean[#] &, Transpose[mlist]];
low = Map[Quantile[#, 0.025] &, Transpose[mlist]];
high = Map[Quantile[#, 1 - 0.025] &, Transpose[mlist]];

PL1 = ListPlot[Z, Joined → True, Frame → True, FrameStyle → Directive[16],
    FrameLabel → {"time t (days)", "log price X(t)"}];
Q3 = ListPlot[mean, Joined → True, PlotRange → All, PlotStyle → Blue];
Q4 = ListPlot[{low, high}, Joined → True,
    PlotRange → All, PlotStyle → {Pink}, Filling → {1 → {2}}];
Show[{PL1, Q3, Q4}, PlotRange → All]
```



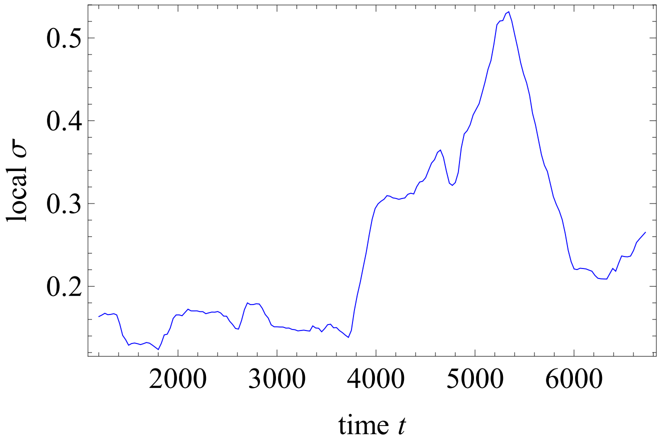
Early warning

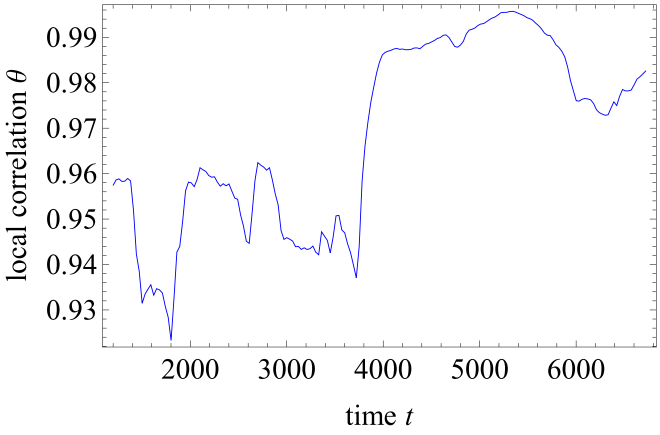
```
σ = 0.02;
r = r0;
tab = {0};
rliste = {r};
Monitor[
  Do[
    data =
     RandomFunction[FractionalBrownianMotionProcess[0.3], {0, 5 * 31 + 1, 1}];
    data = Drop[data["Path"][[All, 2]], 1];
    data = σ (Drop[data, 1] - Drop[data, -1]);
    data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
    ifun  =  Interpolation[data];
    (* xxxxxxxx *)
    r = r0 + v * driver[[30 * t]];
    rliste = Append[rliste, r];
    Q0 = Last[tab];
    s = NDSolve[{Q'[tt] == a * (r - Sqrt[(1 - Q[tt])^2] * Q[tt]) + ifun[tt + 1],
        Q[0] == Q0}, Q, {tt, 0, 30}];
    mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
    tab = Join[tab, mid];
    , {t, 1, 224}];
  , t];
model = L0 + Lrange * tab;

r
0.21788
```

```
θ = EstimatedProcess[model, OrnsteinUhlenbeckProcess[μ, σ, θ]][[3]]
```

```
0.00706933
```

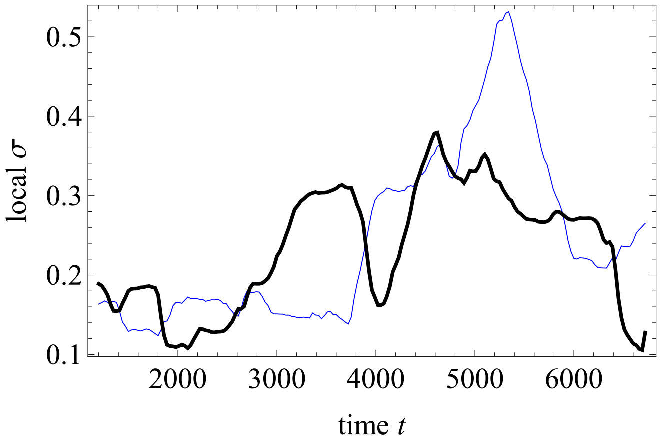
```
win = 1200;
σlist = {};
τlist = {};
Monitor[
  Do[
    qqq = model[[t - win ;; t]];
    σloc = StandardDeviation[qqq]; (*local standard deviation*)
    τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
    (*local correlation scale*)
    σlist = Append[σlist, {t, σloc}];
    τlist = Append[τlist, {t, τloc}];
    , {t, win + 1, Length[model], 30}];
  , t];
QQ3 = ListPlot[σlist, Frame → True, FrameStyle → Directive[16],
   Joined → True, FrameLabel → {"time t", "local σ"}, PlotStyle → Blue];
QQ4 = ListPlot[τlist, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local correlation θ"}, PlotStyle → Blue];
Show[QQ3, PlotRange → All]
Show[QQ4, PlotRange → All]
```

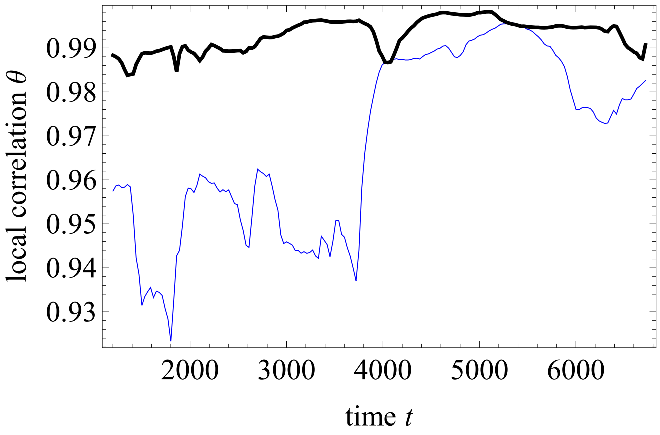




real data

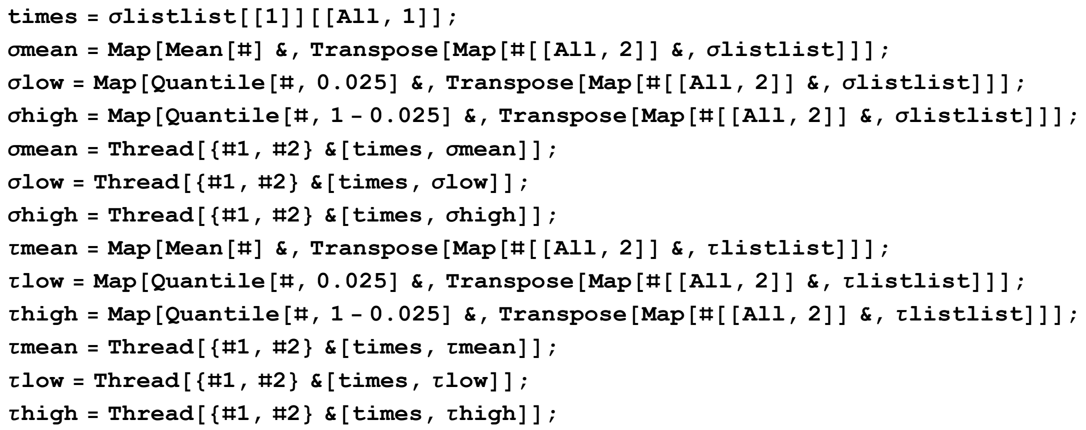
```
win = 1200;
σlist = {};
τlist = {};
Monitor[
  Do[
    qqq = Z[[t - win ;; t]];
    σloc = StandardDeviation[qqq]; (*local standard deviation*)
    τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
    (*local correlation scale*)
    σlist = Append[σlist, {t, σloc}];
    τlist = Append[τlist, {t, τloc}];
    , {t, win + 1, Length[Z], 30}];
  , t];
QQ5 = ListPlot[σlist, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local σ"}, PlotStyle → {Thick, Black}];
QQ6 = ListPlot[τlist, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local correlation θ"}, PlotStyle → {Thick, Black}];
Show[{QQ3, QQ5}, PlotRange → All]
Show[{QQ4, QQ6}, PlotRange → All]
```





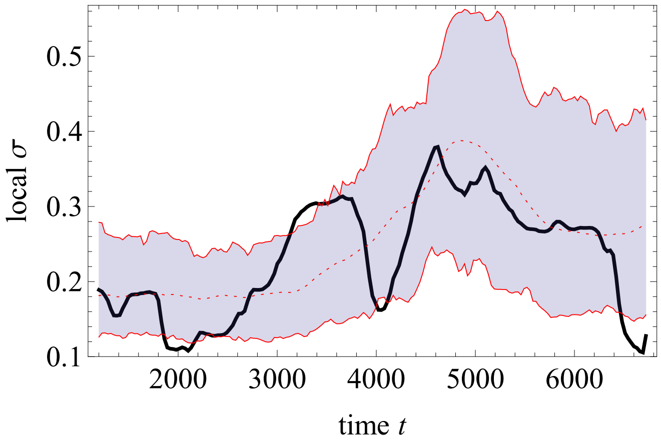
Monte Carlo

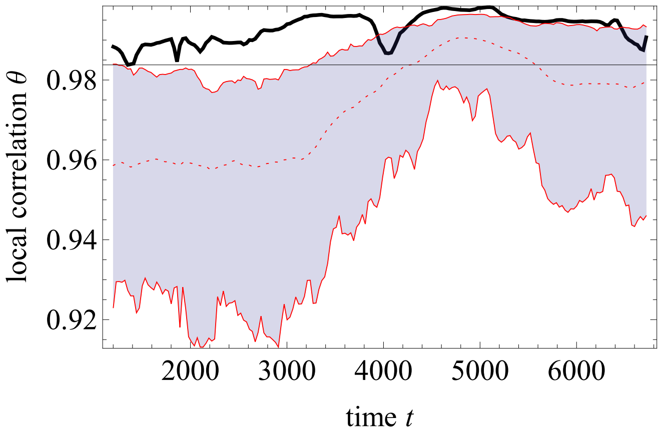
```
τlistlist = {};
σlistlist = {};
Monitor[
  Do[
    σ = 0.02;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       data =
        RandomFunction[FractionalBrownianMotionProcess[.3], {0, 5 * 31 + 1, 1}];
       data = Drop[data["Path"][[All, 2]], 1];
       data = σ (Drop[data, 1] - Drop[data, -1]);
       data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
       ifun = Interpolation[data];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
          Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 224}];
     , t];
    model = L0 + Lrange * tab;
    win = 1200;
    σlist = {};
    τlist = {};
    Monitor[
     Do[
       qqq = model[[t - win ;; t]];
       σloc = StandardDeviation[qqq]; (*local standard deviation*)
       τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
       (*local correlation scale*)
       σlist = Append[σlist, {t, σloc}];
       τlist = Append[τlist, {t, τloc}];
       , {t, win + 1, Length[model], 30}];
     , t];
    τlistlist = Append[τlistlist, τlist];
    σlistlist = Append[σlistlist, σlist];
    , {u, 1, 100}];
  , u];
```



```
times = σlistlist[[1]][[All, 1]];
σmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σmean = Thread[{#1, #2} &[times, σmean]];
σlow = Thread[{#1, #2} &[times, σlow]];
σhigh = Thread[{#1, #2} &[times, σhigh]];
τmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τmean = Thread[{#1, #2} &[times, τmean]];
τlow = Thread[{#1, #2} &[times, τlow]];
τhigh = Thread[{#1, #2} &[times, τhigh]];
```

```
QQQ1 = ListPlot[{σlow, σhigh},
    Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local σ"}, PlotStyle → Red, Filling → {1 → {2}}];
QQQ2 = ListPlot[σmean, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local σ"}, PlotStyle → {Red, Dotted}];
QQQ3 = ListPlot[{τlow, τhigh}, Frame → True, FrameStyle → Directive[16],
    Joined → True, FrameLabel → {"time t", "local σ"},
    PlotStyle → Red, Filling → {1 → {2}}];
QQQ4 = ListPlot[τmean, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local σ"}, PlotStyle → {Red, Dotted}];
Show[{QQ5, QQQ1, QQQ2}, PlotRange → All]
Show[{QQ6, QQQ3, QQQ4}, PlotRange → All]
```





```
Length[Z] / 30.
```

```
224.1
```

Local Hurst exponent

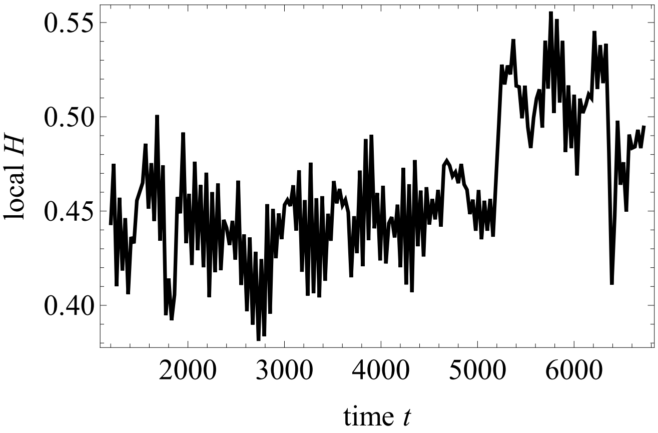
```
fit[z_, o_] := Fit[z, zz^Range[0, o], zz] /. zz → Range[Length[z]];
DFA[x_, o_, imin_, imax_] := (Module[{L, i, X, Z, var, detrends}, var = {};
    Do[L = 2^i;
     X = FoldList[Plus, 0, x - Mean[x]];
     Z = Partition[X, L];
     detrends = Map[# - fit[#, o] &, Z];
     var = Append[var, {L, Sqrt[Mean[Flatten[detrends]^2]]}];,
     {i, imin, imax}];
    Return[var];];)
```

```
Δmodel = Drop[model, 1] - Drop[model, -1];
Fit[Log[DFA[Δmodel, 1, 2, 9]], {zz, 1}, zz]
```

```
-4.16536 + 0.359366 zz
```

```
δt = 1200;
t = δt + 1;
Δmodel = Drop[model, 1] - Drop[model, -1];
H = Fit[Log[DFA[Δmodel[[t - δt ;; t]], 1, 2, 7]], {zz, 1}, zz][[2]] / zz;
```

```
win = 1200;
δt = win;
Hlist = {};
Monitor[
  Do[
    ΔZ = Drop[Z, 1] - Drop[Z, -1];
    qqq = ΔZ[[t - δt ;; t]];

    H = Fit[Log[DFA[qqq, 0, 2, 6]], {zz, 1}, zz][[2]] / zz;

    Hlist = Append[Hlist, {t, H}];
    , {t, win + 1, Length[Z], 30}];
  , t];
QQ3 = ListPlot[Hlist, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local H"}, PlotStyle → {Black, Thick}];
Show[QQ3, PlotRange → All]
```

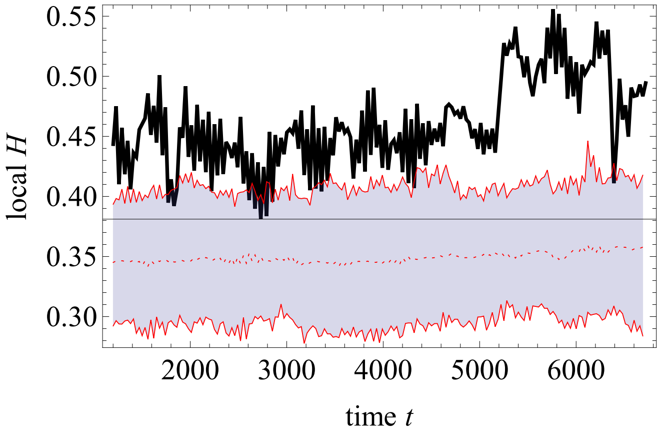


Monte Carlo for local Hurst exponent

```
Hlistlist = {};
Monitor[
  Do[
    σ = 0.02;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       data =
        RandomFunction[FractionalBrownianMotionProcess[0.3], {0, 5 * 31 + 1, 1}];
       data = Drop[data["Path"][[All, 2]], 1];
       data = σ (Drop[data, 1] - Drop[data, -1]);
       data = Thread[{#1, #2} &[Range[5 * 31] / 5., data]];
       ifun = Interpolation[data];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] ⩵ a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
          Q[0] ⩵ Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 224}];
     , t];
    model = L0 + Lrange * tab;
    Δmodel = Drop[model, 1] - Drop[model, -1];
    win = 1200;
    δt = win;
    Hlist = {};
    Monitor[
     Do[
       qqq = Δmodel[[t - δt ;; t]];

       H = Fit[Log[DFA[qqq, 0, 2, 6]], {zz, 1}, zz][[2]] / zz;

       Hlist = Append[Hlist, {t, H}];
       , {t, win + 1, Length[Δmodel], 30}];
     , t];
    Hlistlist = Append[Hlistlist, Hlist];
    , {u, 1, 100}];
  , u];
```

```
times = Hlistlist[[1]][[All, 1]];
Hmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hmean = Thread[{#1, #2} &[times, Hmean]];
Hlow = Thread[{#1, #2} &[times, Hlow]];
Hhigh = Thread[{#1, #2} &[times, Hhigh]];
```

```
QQ5 = ListPlot[{Hlow, Hhigh}, Frame → True, FrameStyle → Directive[16],
   Joined → True, FrameLabel → {"time t", "local \!\(\*
StyleBox[\"H\",\nFontSlant->\"Italic\"]\)"},
   PlotStyle → Red, Filling → {1 → {2}}];
QQ7 = ListPlot[Hmean, Frame → True, FrameStyle → Directive[16],
   Joined → True, FrameLabel → {"time t", "local \!\(\*
StyleBox[\"H\",\nFontSlant->\"Italic\"]\)"}, PlotStyle → {Red, Dotted}];
Show[{QQ3, QQ5, QQ7}, PlotRange → All]
```

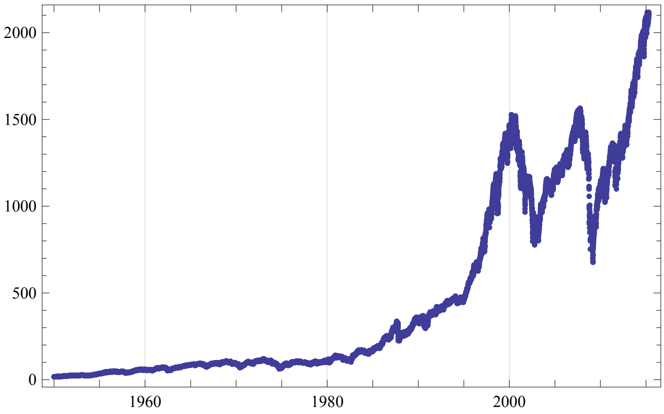


Appendix F

Source code of S&P 500 index analysis

This appendix includes analysis of stock market prices for 1950-2015. We use random walk process and find the early warning signals.

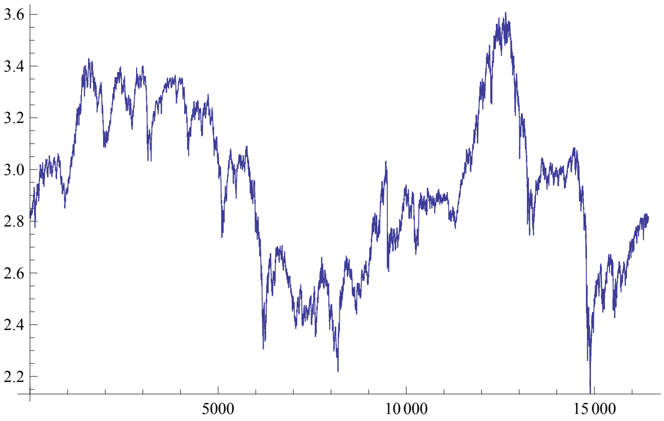
```
DateListPlot[FinancialData["SP500", All]]
x = FinancialData["SP500", All, "Value"];
```



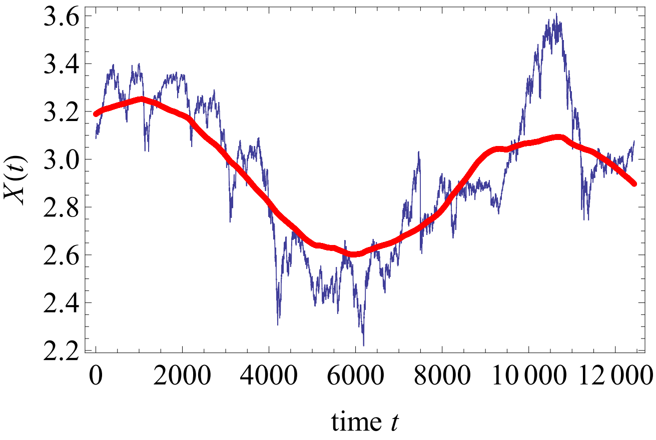
```
Length[x]
```

```
16 436
```

```
y = Log[x];
μ = Mean[Drop[y, 1] - Drop[y, -1]];
y = y - μ * Range[Length[y]];
ListPlot[y, Joined → True]
```

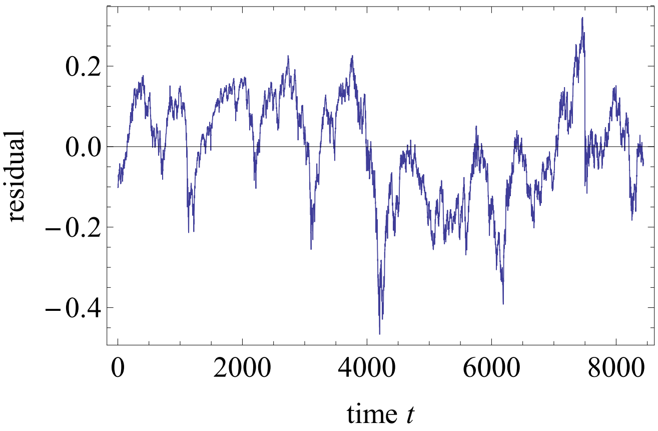


```
win = 4000;
L = MovingAverage[y, win];
PL1 = ListPlot[Drop[Drop[y, win / 2], -win / 2], Joined → True, Frame → True,
    FrameStyle → Directive[16], FrameLabel → {"time t", "X(t)"} ];
PL2 = ListPlot[L, PlotStyle → Red];
residual = Drop[Drop[y, win / 2 - 1], -win / 2] - L;
PL0 = Show[PL1, PL2]
```



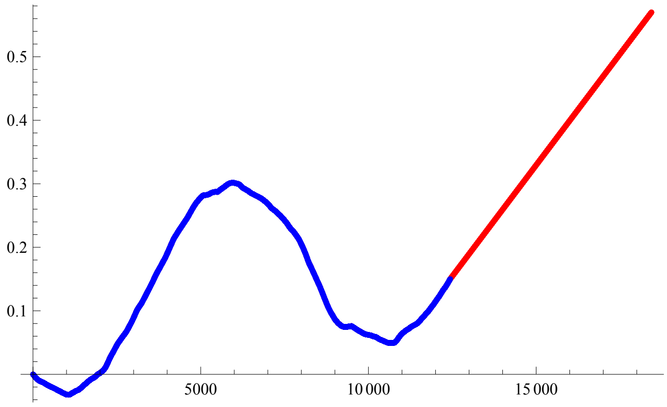
Residual before the crisis :

```
ListPlot[Drop[residual, -4000],
 Joined → True, Frame → True, FrameStyle → Directive[16],
 FrameLabel → {"time t", "residual"}, PlotRange → All ]
```



```
θ = EstimatedProcess[Drop[residual, -4000],
     OrnsteinUhlenbeckProcess[μ, σ, θ]][[3]];
Clear[a]
sol = Solve[a Sqrt[1 - 4 * 0.14] == θ, a];
a = sol[[1]][[1]][[2]]
0.00370784
```

```
L0 = First[L];
Lrange = Last[L] - First[L];
Lrange = Lrange / 0.15;
driver = (L - L0) / Lrange;
r0 = 0.0;
v = 1;
PL0 = ListPlot[r0 + v * driver, PlotStyle → Blue];
driver = Join[driver, Last[driver] + 0.00007 * Range[6000]];
PL1 = ListPlot[r0 + v * driver, PlotStyle → Red];
Show[PL1, PL0]
```

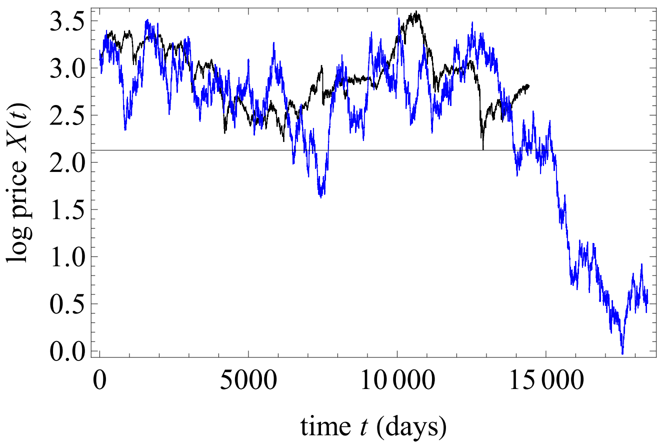


```
Length[driver] / 30.
```

```
614.533
```

```
σ = 0.025;
r = r0;
tab = {0};
rliste = {r};
Monitor[
 Do[
   rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
   rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
   ifun = Interpolation[rand];
   (* xxxxxxxx *)
   r = r0 + v * driver[[30 * t]];
   rliste = Append[rliste, r];
   Q0 = Last[tab];
   s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt])^2] * Q[tt]) + ifun[tt + 1],
      Q[0] == Q0}, Q, {tt, 0, 30}];
   mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
   tab = Join[tab, mid];
   , {t, 1, 614}];
 , t]
pricemodel = tab;
```

```
PL1 = ListPlot[Drop[Drop[y, win / 2], 0],
   Joined → True, Frame → True, FrameStyle → Directive[16],
   FrameLabel → {"time  t  (days)", "log price  X(t)"} , PlotStyle → Black];
Q2 = ListPlot[L0 + Lrange ∗ pricemodel, Joined → True,
   PlotRange → All, PlotStyle → Blue];
Show[{PL1, Q2}, PlotRange → All]
```



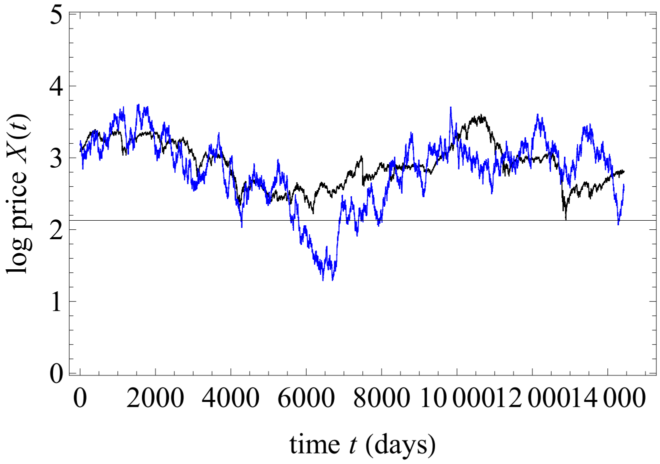
```
Z = Drop[Drop[y, win / 2], 0];
```

```
Length[Z] / 30.
```

```
481.167
```

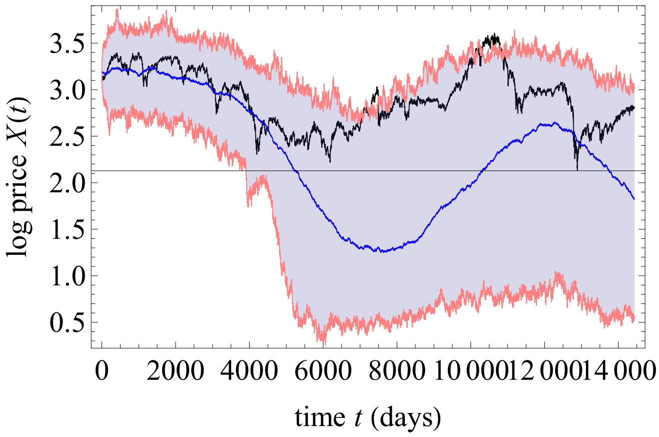
```
σ = 0.025;
r = r0;
tab = {0};
rliste = {r};
Monitor[
 Do[
   rand = RandomReal[NormalDistribution[0, σ], 5 ∗ 31];
   rand = Thread[{#1, #2} &[Range[5 ∗ 31] / 5., rand]];
   ifun  =  Interpolation[rand];
   (∗ xxxxxxxx ∗)
   r = r0 + v ∗ driver[[30 ∗ t]];
   rliste = Append[rliste, r];
   Q0 = Last[tab];
   s = NDSolve[{Q′[tt] ⩵ a ∗ (r - Sqrt[(1 - Q[tt])^2] ∗ Q[tt]) + ifun[tt + 1],
      Q[0] ⩵ Q0}, Q, {tt, 0, 30}];
   mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
   tab = Join[tab, mid];
   , {t, 1, 481}];
 , t]
model = L0 + Lrange ∗ tab;
```

```
PL1 = ListPlot[Z, Joined → True, Frame → True, FrameStyle → Directive[16],
   FrameLabel → {"time  t  (days)", "log price  X(t)"}, PlotStyle → Black ];
Q2 = ListPlot[model, Joined → True, PlotRange → All, PlotStyle → Blue];
Show[{PL1, Q2}, PlotRange → {{0, 15 000}, {0, 5}}]
```



```
mlist = {};
Monitor[
  Do[
    σ = 0.025;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
       rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
       ifun  = Interpolation[rand];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
           Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 481}];
     , t];
    model = L0 + Lrange * tab;
    mlist = Append[mlist, model];
    , {j, 1, 100}];
  , j];
```

```
mean = Map[Mean[#] &, Transpose[mlist]];
low = Map[Quantile[#, 0.025] &, Transpose[mlist]];
high = Map[Quantile[#, 1 - 0.025] &, Transpose[mlist]];

PL1 = ListPlot[Z, Joined → True, Frame → True, FrameStyle → Directive[16],
    FrameLabel → {"time  t  (days)", "log price  X(t)"} , PlotStyle → Black];
Q3 = ListPlot[mean, Joined → True, PlotRange → All, PlotStyle → Blue];
Q4 = ListPlot[{low, high}, Joined → True,
    PlotRange → All, PlotStyle → {Pink}, Filling → {1 → {2}}];
Show[{PL1, Q3, Q4}, PlotRange → All]
```



Early warning

```
σ = 0.025;
r = r0;
tab = {0};
rliste = {r};
Monitor[
  Do[
    rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
    rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
    ifun  =  Interpolation[rand];
    (*  xxxxxxxx  *)
    r = r0 + v * driver[[30 * t]];
    rliste = Append[rliste, r];
    Q0 = Last[tab];
    s = NDSolve[{Q'[tt] == a * (r - Sqrt[(1 - Q[tt])^2] * Q[tt]) + ifun[tt + 1],
        Q[0] == Q0}, Q, {tt, 0, 30}];
    mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
    tab = Join[tab, mid];
    , {t, 1, 481}];
  , t];
model = L0 + Lrange * tab;
```

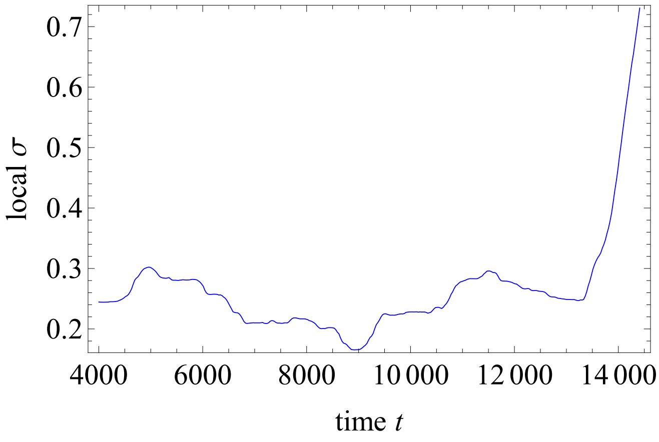
```
r
```

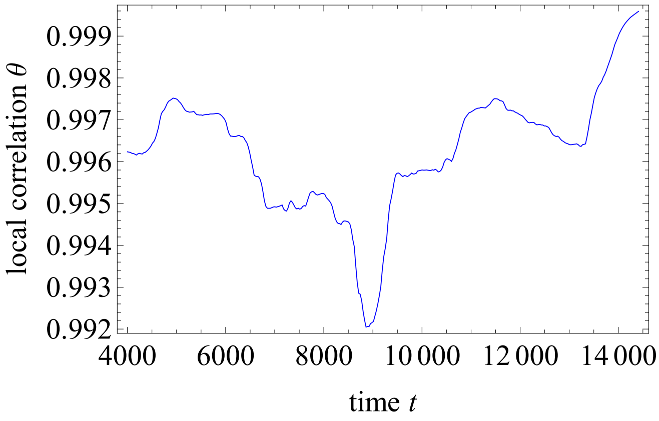
```
0.28951
```

```
θ = EstimatedProcess[model, OrnsteinUhlenbeckProcess[μ, σ, θ]][[3]]
```

```
0.0024595
```

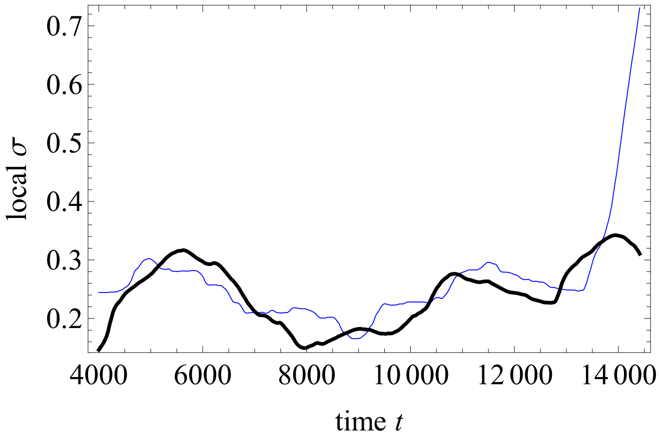
```
win = 4000;
σlist = {};
τlist = {};
Monitor[
  Do[
    qqq = model[[t - win ;; t]];
    σloc = StandardDeviation[qqq]; (*local standard deviation*)
    τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
    (*local correlation scale*)
    σlist = Append[σlist, {t, σloc}];
    τlist = Append[τlist, {t, τloc}];
    , {t, win + 1, Length[model], 30}];
  , t];
QQ3 = ListPlot[σlist, Frame → True, FrameStyle → Directive[16],
   Joined → True, FrameLabel → {"time t", "local σ"}, PlotStyle → Blue];
QQ4 = ListPlot[τlist, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local correlation θ"}, PlotStyle → Blue];
Show[QQ3, PlotRange → All]
Show[QQ4, PlotRange → All]
```

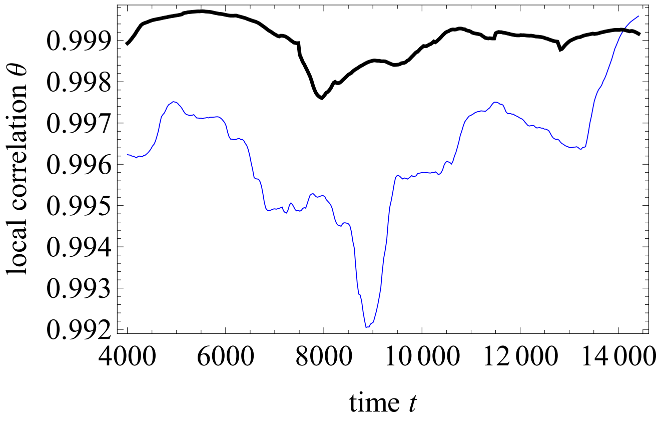




real data:

```
win = 4000;
σlist = {};
τlist = {};
Monitor[
  Do[
    qqq = Z[[t - win ;; t]];
    σloc = StandardDeviation[qqq]; (*local standard deviation*)
    τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
    (*local correlation scale*)
    σlist = Append[σlist, {t, σloc}];
    τlist = Append[τlist, {t, τloc}];
    , {t, win + 1, Length[Z], 30}];
  , t];
QQ5 = ListPlot[σlist, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local σ"}, PlotStyle → {Thick, Black}];
QQ6 = ListPlot[τlist, Frame → True, FrameStyle → Directive[16], Joined → True,
    FrameLabel → {"time t", "local correlation θ"}, PlotStyle → {Thick, Black}];
Show[{QQ3, QQ5}, PlotRange → All]
Show[{QQ4, QQ6}, PlotRange → All]
```

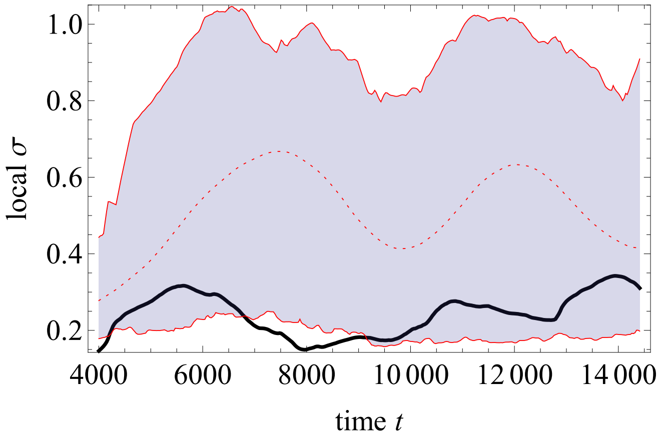


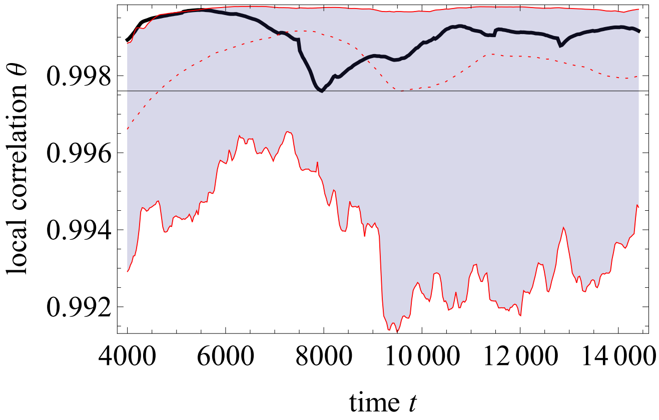


monte carlo

```
τlistlist = {};
σlistlist = {};
Monitor[
  Do[
    σ = 0.025;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
       rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
       ifun = Interpolation[rand];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
          Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 481}];
     , t];
    model = L0 + Lrange * tab;
    win = 4000;
    σlist = {};
    τlist = {};
    Monitor[
     Do[
       qqq = model[[t - win ;; t]];
       σloc = StandardDeviation[qqq]; (*local standard deviation*)
       τloc = Correlation[Drop[qqq, 1], Drop[qqq, -1]];
       (*local correlation scale*)
       σlist = Append[σlist, {t, σloc}];
       τlist = Append[τlist, {t, τloc}];
       , {t, win + 1, Length[model], 30}];
     , t];
    τlistlist = Append[τlistlist, τlist];
    σlistlist = Append[σlistlist, σlist];
    , {u, 1, 100}];
  , u];
```

```
times = σlistlist[[1]][[All, 1]];
σmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, σlistlist]]];
σmean = Thread[{#1, #2} &[times, σmean]];
σlow = Thread[{#1, #2} &[times, σlow]];
σhigh = Thread[{#1, #2} &[times, σhigh]];
τmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, τlistlist]]];
τmean = Thread[{#1, #2} &[times, τmean]];
τlow = Thread[{#1, #2} &[times, τlow]];
τhigh = Thread[{#1, #2} &[times, τhigh]];

QQQ1 = ListPlot[{σlow, σhigh},
   Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local σ"}, PlotStyle → Red, Filling → {1 → {2}}];
QQQ2 = ListPlot[σmean, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local σ"}, PlotStyle → {Red, Dotted}];
QQQ3 = ListPlot[{τlow, τhigh}, Frame → True, FrameStyle → Directive[16],
   Joined → True, FrameLabel → {"time t", "local σ"},
   PlotStyle → Red, Filling → {1 → {2}}];
QQQ4 = ListPlot[τmean, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local σ"}, PlotStyle → {Red, Dotted}];
Show[{QQ5, QQQ1, QQQ2}, PlotRange → All]
Show[{QQ6, QQQ3, QQQ4}, PlotRange → All]
```





Local Hurst exponent

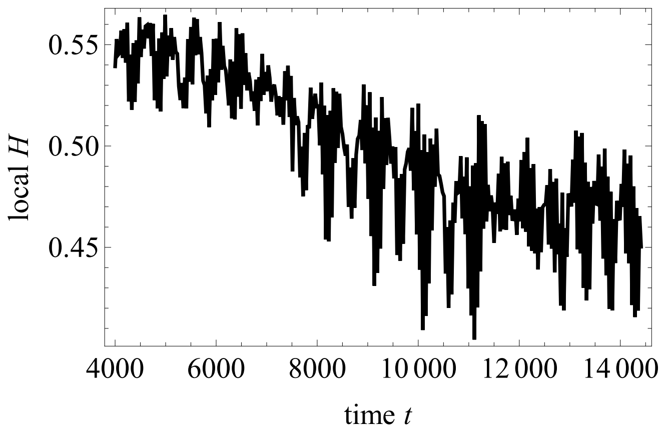
```
fit[z_, o_] := Fit[z, zz^Range[0, o], zz] /. zz → Range[Length[z]];
DFA[x_, o_, imin_, imax_] := (Module[{L, i, X, Z, var, detrends}, var = {};
    Do[L = 2^i;
     X = FoldList[Plus, 0, x - Mean[x]];
     Z = Partition[X, L];
     detrends = Map[# - fit[#, o] &, Z];
     var = Append[var, {L, Sqrt[Mean[Flatten[detrends]^2]]}];,
     {i, imin, imax}];
    Return[var];];)
```

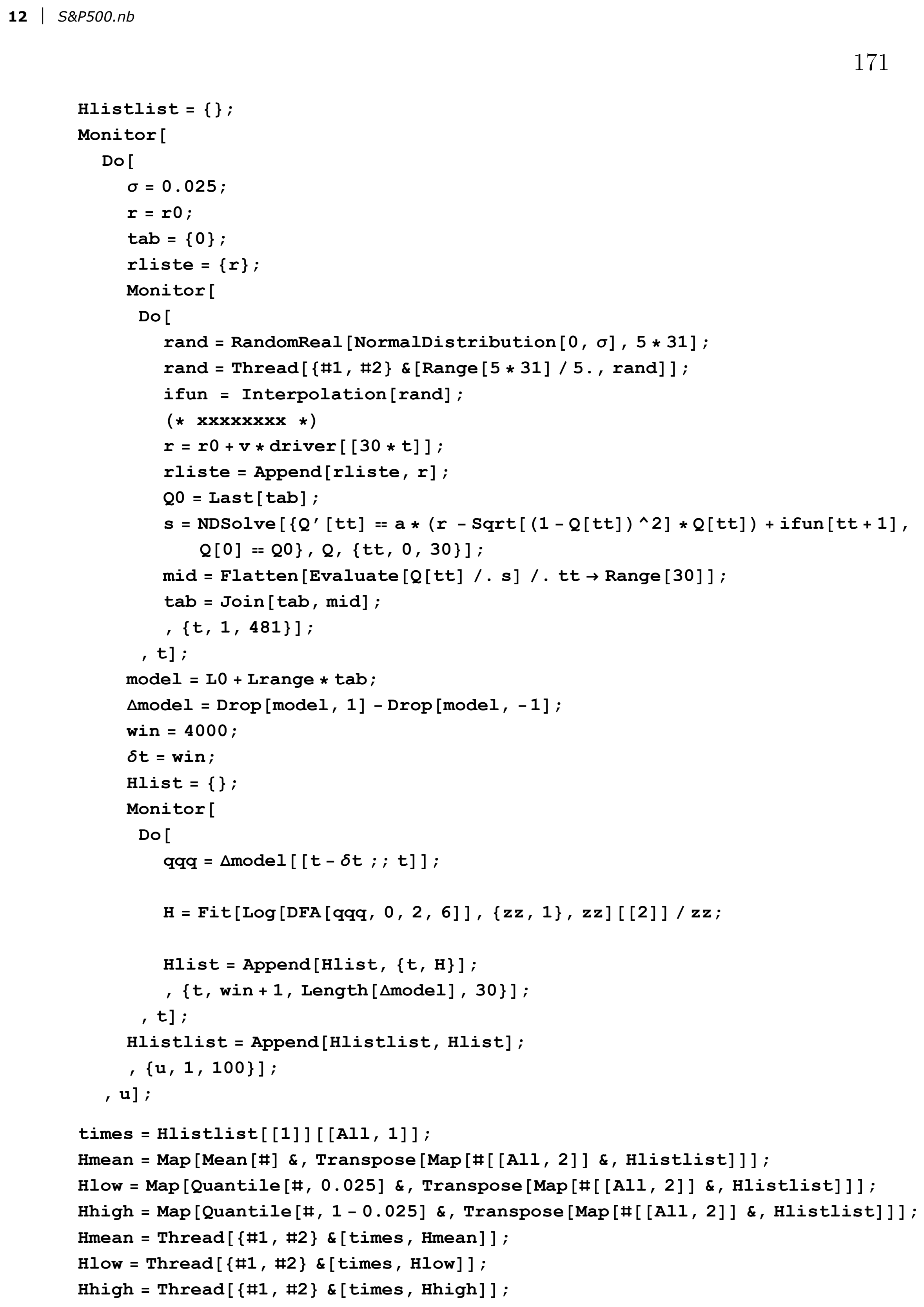
```
Δmodel = Drop[model, 1] - Drop[model, -1];
Fit[Log[DFA[Δmodel, 1, 2, 9]], {zz, 1}, zz]
```

```
-5.29805 + 0.52037 zz
```

```
δt = 4000;
t = δt + 1;
Δmodel = Drop[model, 1] - Drop[model, -1];
H = Fit[Log[DFA[Δmodel[[t - δt ;; t]], 1, 2, 7]], {zz, 1}, zz][[2]] / zz;
```

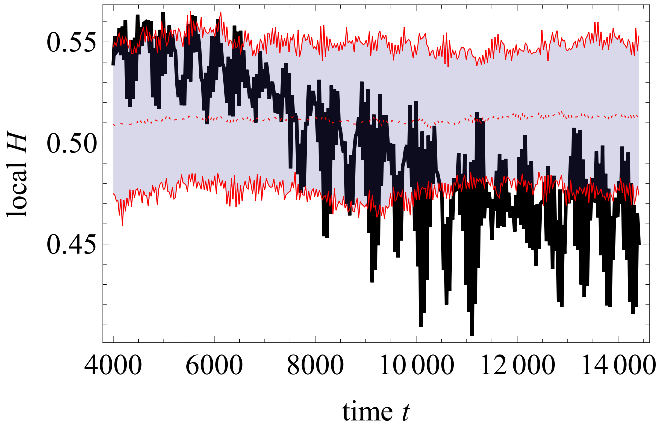
```
win = 4000;
δt = win;
Hlist = {};
Monitor[
  Do[
    ΔZ = Drop[Z, 1] - Drop[Z, -1];
    qqq = ΔZ[[t - δt ;; t]];

    H = Fit[Log[DFA[qqq, 0, 2, 6]], {zz, 1}, zz][[2]] / zz;

    Hlist = Append[Hlist, {t, H}];
    , {t, win + 1, Length[Z], 30}];
  , t];
QQ3 = ListPlot[Hlist, Frame → True, FrameStyle → Directive[16], Joined → True,
   FrameLabel → {"time t", "local H"}, PlotStyle → {Black, Thick}];
Show[QQ3, PlotRange → All]
```





```
Hlistlist = {};
Monitor[
  Do[
    σ = 0.025;
    r = r0;
    tab = {0};
    rliste = {r};
    Monitor[
     Do[
       rand = RandomReal[NormalDistribution[0, σ], 5 * 31];
       rand = Thread[{#1, #2} &[Range[5 * 31] / 5., rand]];
       ifun = Interpolation[rand];
       (* xxxxxxxx *)
       r = r0 + v * driver[[30 * t]];
       rliste = Append[rliste, r];
       Q0 = Last[tab];
       s = NDSolve[{Q′[tt] == a * (r - Sqrt[(1 - Q[tt]) ^2] * Q[tt]) + ifun[tt + 1],
          Q[0] == Q0}, Q, {tt, 0, 30}];
       mid = Flatten[Evaluate[Q[tt] /. s] /. tt → Range[30]];
       tab = Join[tab, mid];
       , {t, 1, 481}];
     , t];
    model = L0 + Lrange * tab;
    Δmodel = Drop[model, 1] - Drop[model, -1];
    win = 4000;
    δt = win;
    Hlist = {};
    Monitor[
     Do[
       qqq = Δmodel[[t - δt ;; t]];

       H = Fit[Log[DFA[qqq, 0, 2, 6]], {zz, 1}, zz][[2]] / zz;

       Hlist = Append[Hlist, {t, H}];
       , {t, win + 1, Length[Δmodel], 30}];
     , t];
    Hlistlist = Append[Hlistlist, Hlist];
    , {u, 1, 100}];
  , u];

times = Hlistlist[[1]][[All, 1]];
Hmean = Map[Mean[#] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hlow = Map[Quantile[#, 0.025] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hhigh = Map[Quantile[#, 1 - 0.025] &, Transpose[Map[#[[All, 2]] &, Hlistlist]]];
Hmean = Thread[{#1, #2} &[times, Hmean]];
Hlow = Thread[{#1, #2} &[times, Hlow]];
Hhigh = Thread[{#1, #2} &[times, Hhigh]];
```

```
QQ5 = ListPlot[{Hlow, Hhigh}, Frame → True, FrameStyle → Directive[16],
   Joined → True, FrameLabel → {"time t", "local \!\(\*
StyleBox[\"H\",\nFontSlant->\"Italic\"]\)"},
   PlotStyle → Red, Filling → {1 → {2}}];
QQ7 = ListPlot[Hmean, Frame → True, FrameStyle → Directive[16],
   Joined → True, FrameLabel → {"time t", "local \!\(\*
StyleBox[\"H\",\nFontSlant->\"Italic\"]\)"}, PlotStyle → {Red, Dotted}];
Show[{QQ3, QQ5, QQ7}, PlotRange → All]
```



UNIVERSITY OF TROMSØ