Prediction of Oil Prices Turning Points with LPPL

B. Dufour J. Mourgues-Haroche T. Charbonnier A. Remiat

15 janvier 2025

Introduction to financial bubbles modeling

Crude oil is crucial to the global economy, with price volatility causing major economic and political disruptions. Accurately predicting price turning points is essential for strategic reserves and investment.
This paper presents an enhanced LPPL model, optimized via a multi-population genetic algorithm (MPGA), for improved forecasting accuracy.

Keywords of the study

- Financial bubbles detection
- Non-linear models estimation
- Heuristic & Genetic algorithms

Context of LPPL Models

- LPPL models aim to describe speculative bubbles in financial markets.
- **Key Assumption :** A bubble exhibits super-exponential growth with log-periodic oscillations as it approaches a critical point.
- **Motivation :** To capture the interplay between rational traders and noise traders in destabilizing the market.

Historical Use

- Empirical validation on commodities and equity indices.
- Interesting extensions on currencies and cryptocurrencies

Traders interplay : The Ising model

Each trader's position ($s_i = +1$ for long, $s_i = -1$ for short) is determined by :

$$s_i = \text{sign}\left(\sum_{j \in N(i)} K s_j + \sigma \epsilon_i + G \right),$$

where :

- $N(i)$: Trader i's network.
- $K > 0$: Coupling strength (influence of peers).
- $\sigma > 0$: Strength of idiosyncratic factors.
- $\epsilon_i \sim N(0, 1)$: Independent random shock.
- G : Global influence.

Agent-Based Foundations of LPPL Models

LPPL models are derived from agent-based interactions between :

- **Noise traders :** Exhibit herding behavior, amplifying trends.
- **Rational traders :** Attempt to exploit mispricings but may fail to stabilize the market.

Conditional Crash Probability

The conditional probability of a crash, $h(t)$, given that the bubble has not yet burst, represents the likelihood of a sudden collective sell-off :

$$h(t) = B(t_c - t)^{-b}, \quad b \in (0, 1).$$

This describes the increasing risk of market makers failing to absorb shocks as t approaches t_c.

Log-Periodic Power Law Equation

General LPPL Form

$$p(t) = A + B(T_c - t)^{\alpha} + C(T_c - t)^{\alpha} \cos\left(\omega \ln(T_c - t) + \phi\right)$$

- $p(t)$: Asset price or market indicator.
- t_c : Critical time (crash or turning point).
- A, B, C, ω, ϕ : Other parameters to be estimated.
- α : Critical exponent $(0 < \alpha < 1)$.

Interpretation

- Feedback loops create instability, while oscillations arise from trader interactions.
- Explains market crashes as endogenous events, not purely external shocks.

Derivation of linear parameters

- Linear parameters can be directly derived using a least squares approach for given nonlinear parameters :
- Definitions :

$$f_j = (T_c - t_j)^{\alpha} \quad \text{and} \quad g_j = (T_c - t_j)^{\alpha} \cos(\omega \ln(T_c - t_j) + \phi)$$

$$\begin{bmatrix} A \\ B \\ C \end{bmatrix} = \left(\mathrm{V}^T \mathrm{V}\right)^{-1} \mathrm{V}^T \mathrm{Y}$$

where :

$$\mathrm{V} = \begin{bmatrix} 1 & f_1 & g_1 \\ 1 & f_2 & g_2 \\ \vdots & \vdots & \vdots \\ 1 & f_J & g_J \end{bmatrix}, \quad \mathrm{Y} = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(J) \end{bmatrix}$$

Lomb Periodogram Analysis in LPPL Models

Purpose of the Analysis

- The Lomb periodogram is used to validate the turning points (t_c) predicted by the LPPL model.
- It checks if the frequency $\frac{\omega}{2\pi}$ derived from the LPPL fits matches the frequency of the periodic oscillations near the critical time t_c.

Why the Lomb Periodogram ?

- Handles **non-uniform time series**, common in financial data.
- Provides an **objective method** for identifying periodic components.
- Effective for detecting periodic oscillations in noisy environments.

Lomb Periodogram Method

Power Spectral Density

The power spectral density $P(f)$ is computed for the frequency series using the formula :

$$P(f) = \frac{1}{2\sigma^2}\left[\frac{\left(\sum_{j=1}^{J}(x_j - \bar{x})\cos\left(2\pi f(t_j - \tau)\right)\right)^2}{\sum_{j=1}^{J}\cos^2\left(2\pi f(t_j - \tau)\right)} + \frac{\left(\sum_{j=1}^{J}(x_j - \bar{x})\sin\left(2\pi f(t_j - \tau)\right)\right)^2}{\sum_{j=1}^{J}\sin^2\left(2\pi f(t_j - \tau)\right)}\right]$$

Where :

- $x_j = y_j - A - B(t_c - t_j)^a$
- $\bar{x}$ is the mean of x_j
- σ^2 is the variance of x_j.

Frequency Calculation in Lomb Periodogram

Offset time

The time offset t is calculated as :

$$t = \frac{1}{4\pi f} \arctan\left(\sum_{j=1}^{J} \sin(4\pi f t_j) \Big/ \sum_{j=1}^{J} \cos(4\pi f t_j) \right)$$

- With $tj = ln(t_c - t)^a$

Key Process

- Invalid frequencies in $P(f)$ are removed (e.g., values below a threshold or the most probable frequency).
- If no valid values remain, the predicted turning points are **rejected**.
- Otherwise, the frequency with the **maximum valid value** is used.

Validation of Turning Points with Lomb Periodogram

Validation Criteria

- The frequency from the Lomb periodogram is compared to $\frac{\omega}{2\pi}$ from MPGA.
- A turning point is valid if the difference between frequencies is less than 0.3.

Conclusion

- Only turning points passing the Lomb periodogram test are **recorded as valid**.
- Ensures robustness in predicting critical times (t_c) in LPPL analysis.

Why Heuristic Algorithms for LPPL ?

Non-Linear Nature of LPPL

- The LPPL model is highly **non-linear**, with parameters like ω, t_c, and m interacting in complex ways.
- Standard estimation methods (e.g., least squares, likelihood) fail to estimate the model.

Heuristic Algorithms

- **Definition :** Algorithms inspired by natural processes or optimization heuristics to explore large and complex solution spaces.
- Designed to :
  - Avoid local minima.
  - Handle high-dimensional, non-linear models effectively.
- Examples : Genetic algorithms, simulated annealing, particle swarm optimization.

Genetic Algorithms : Principles

Key Concepts :

- Inspired by **natural evolution** and genetic selection.
- Operates on a population of candidate solutions, evolving them through :
  - **Selection :** Fittest individuals are chosen for reproduction.
  - **Crossover :** Genetic material is combined to create offspring.
  - **Mutation :** Small random changes introduce diversity.

Why GAs for LPPL ?

- Effectively searches non-linear, multi-dimensional parameter spaces.
- Balances **exploration** (diverse solutions) and **exploitation** (refining good solutions).

Multi-Population Genetic Algorithm (MPGA)

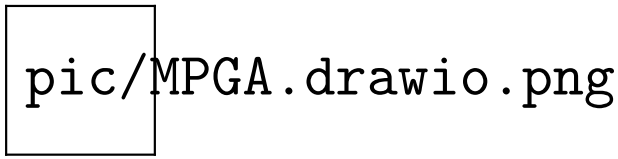
Principle of MPGA

- MPGA extends standard genetic algorithms by introducing **multiple sub-populations**.
- Each sub-population evolves independently, avoiding premature convergence to local minima.
- Periodic **migration of individuals** between populations introduces diversity and improves global search.

Advantages for LPPL Estimation

- **Avoids local minima :** Independent sub-populations explore different areas of the parameter space.
- **Improved convergence :** Migration ensures diversity and prevents stagnation.
- **Effective for LPPL :** Handles the complex, multi-modal structure of the LPPL model.

MPGA Process



Particle Swarm Optimization (PSO)

Principle of PSO

- PSO is inspired by **social behavior of swarms** (e.g., birds).
- It optimizes a function by iteratively moving a population of particles (candidate solutions) through the search space.
- Each particle adjusts its position based on :
  - **Personal Best (p_i) :** Best solution it has found so far.
  - **Global Best (g) :** Best solution found in the entire swarm.

PSO Algorithm Steps

1. Initialize particles with random positions and velocities.
2. Evaluate the fitness of each particle.
3. Update velocities and positions :

$$v_i(t+1) = wv_i(t) + c_1 r_1(p_i - x_i) + c_2 r_2(g - x_i)$$

$$x_i(t+1) = x_i(t) + v_i(t+1)$$

Simulated Annealing (SA)

Principle of SA

- Simulated Annealing (SA) is inspired by the **annealing process** in metallurgy : Material is heated and slowly cooled to remove defects and reach a stable structure.
- SA mimics this by exploring the search space and accepting worse solutions to escape local minima.

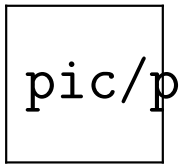
SA Algorithm Steps

1. Start with an initial solution and a high "temperature" T.
2. Generate a new candidate by making a small random change.
3. Evaluate the fitness of the new solution :
   - If better, accept it, else accept it with probability :

$$P = \exp\left(-\frac{\Delta E}{T}\right),$$

where ΔE is the fitness difference.

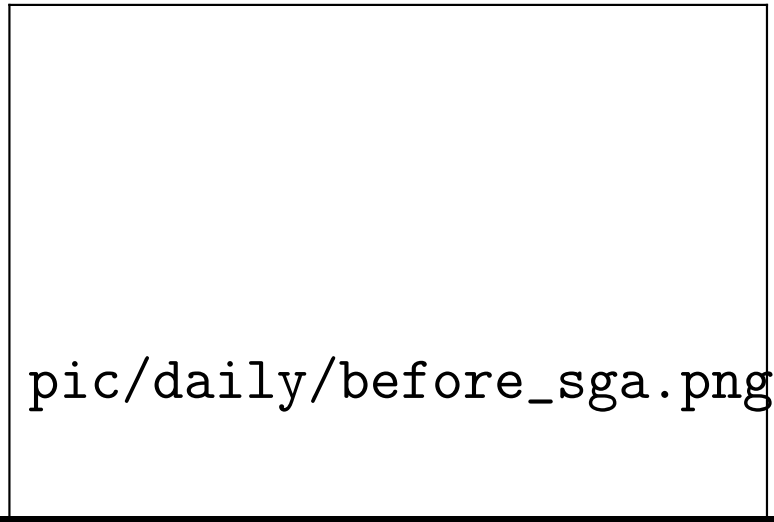
Turning points prediction in practice



Framework Organization


pic/daily/before_mpga.png

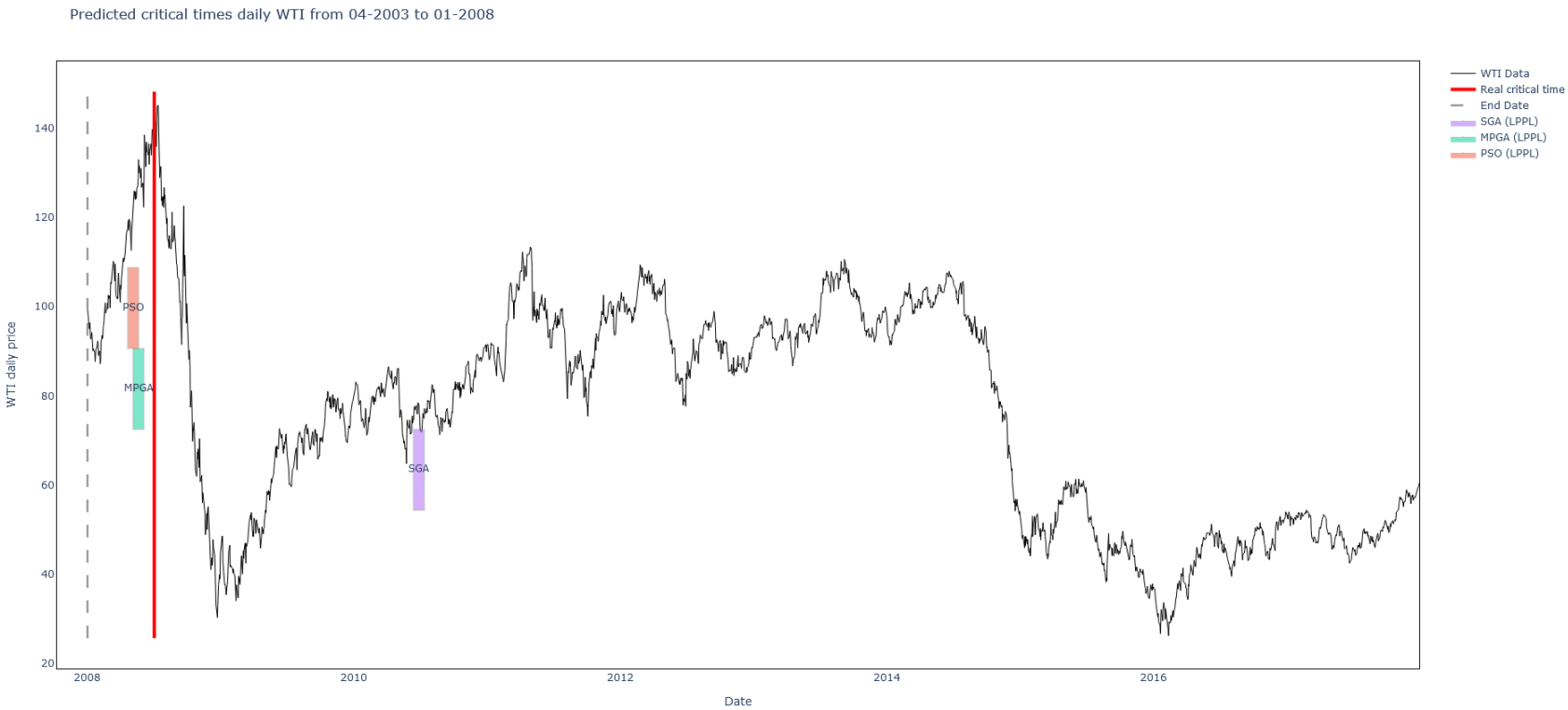

pic/daily/before_PSO.png


pic/daily/before_sga.png


pic/daily/before_sa.png

Comparison of Prediction Results (daily)

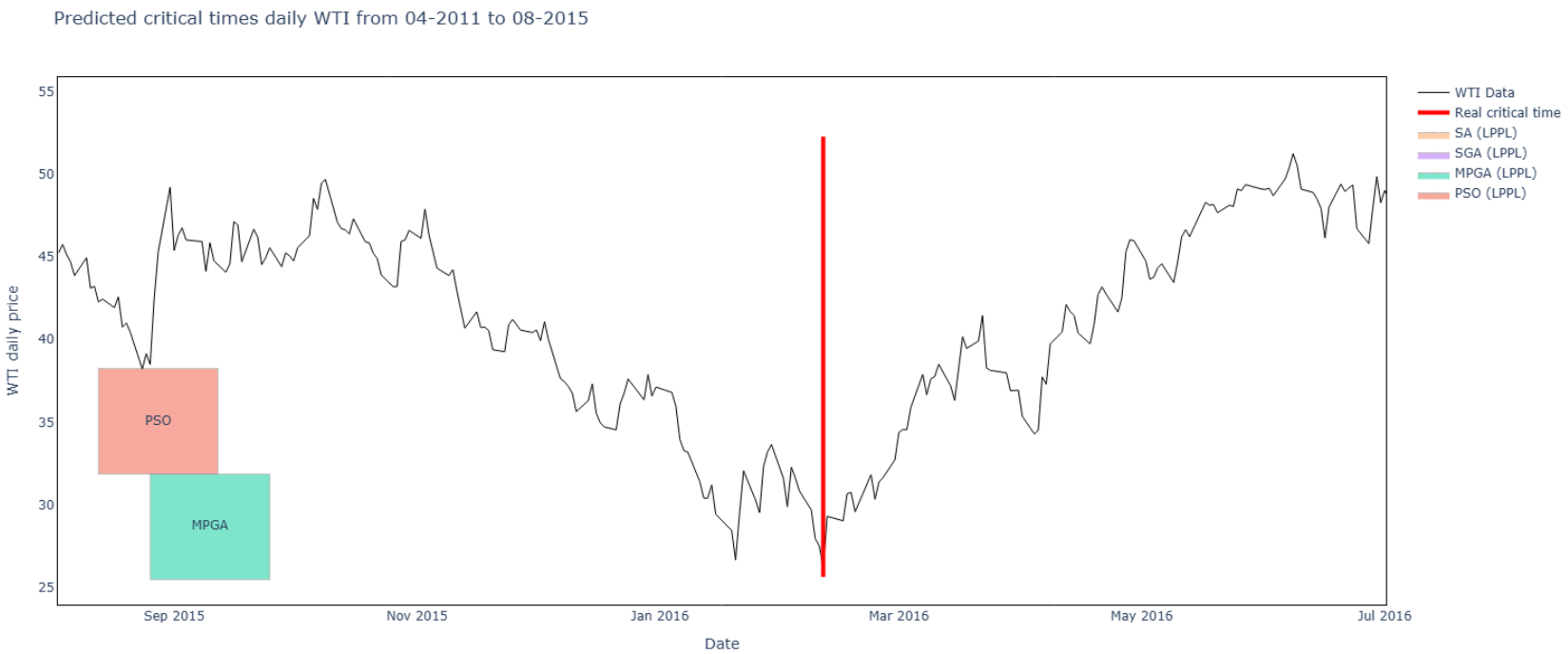
Comparison of Prediction Results (daily)



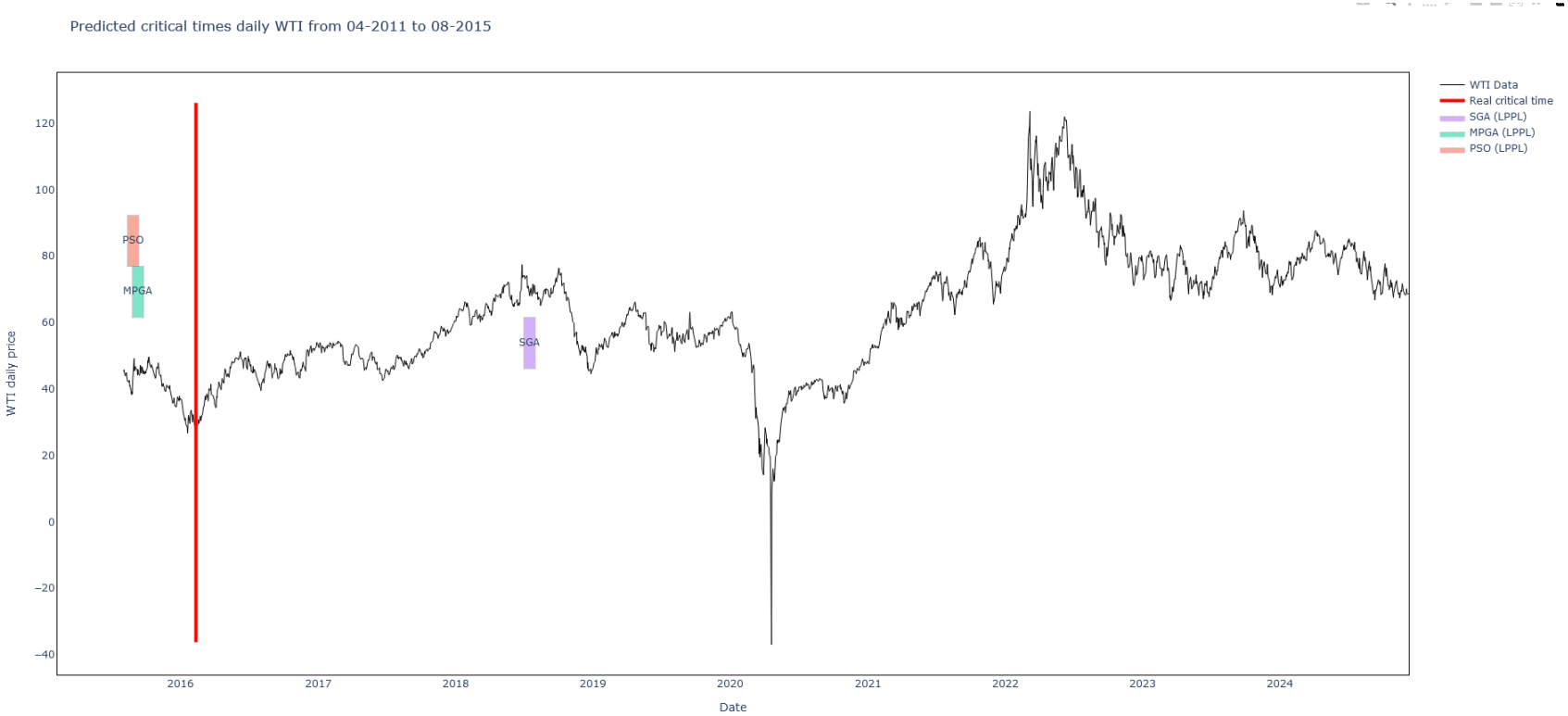
Comparison of Prediction Results (daily)

Comparison of Prediction Results (daily)

Comparison of Prediction Results (daily)



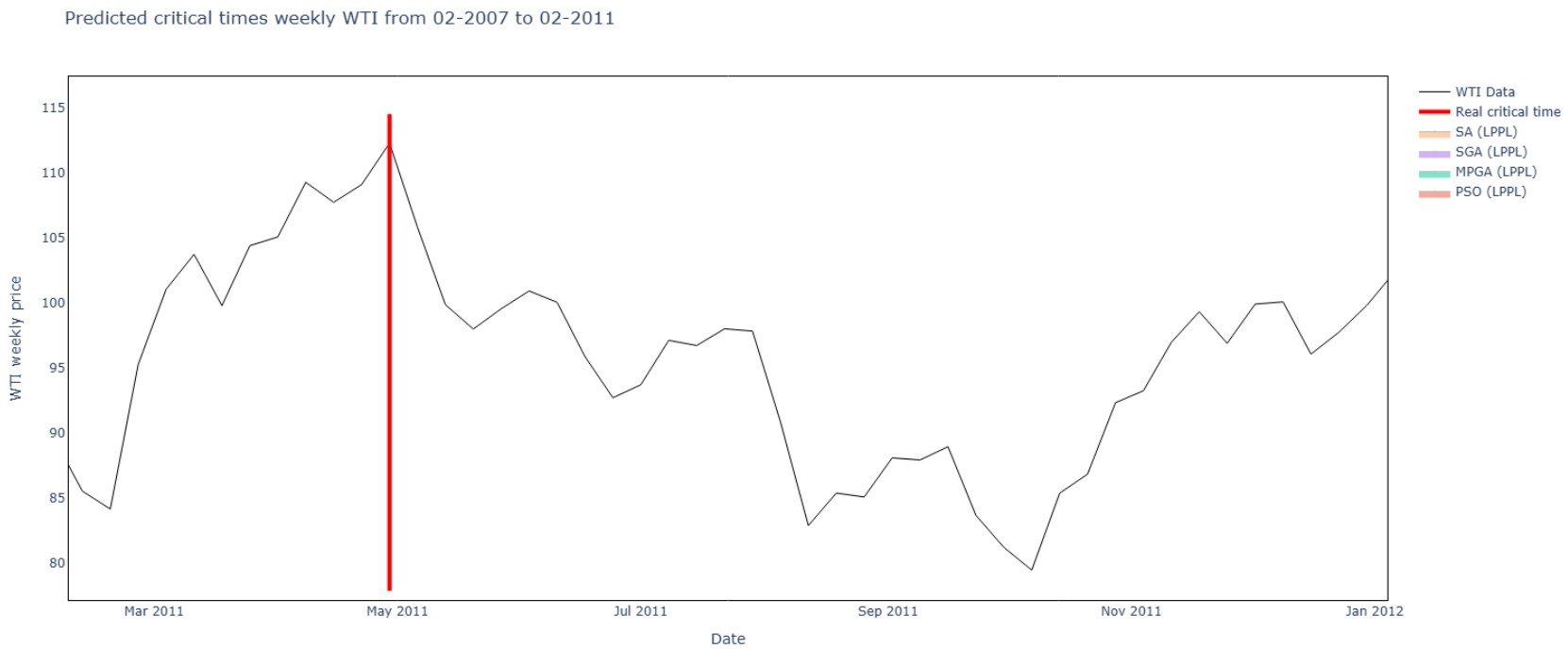
Comparison of Prediction Results (daily)



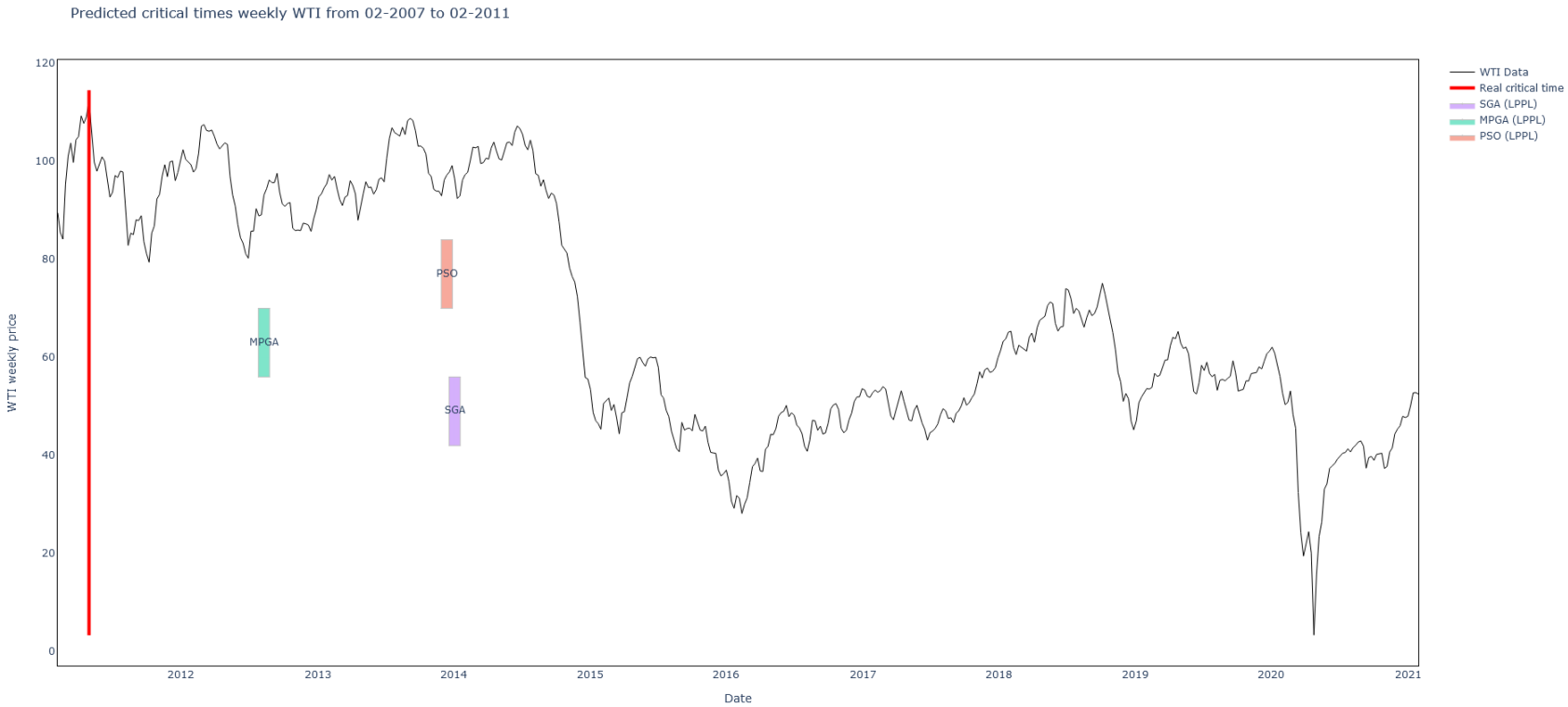
Comparison of Prediction Results (weekly)

Comparison of Prediction Results (weekly)

Comparison of Prediction Results (weekly)



Comparison of Prediction Results (weekly)



Comparison of Prediction Results (weekly)

Comparison of Prediction Results (weekly)

Comparison of Prediction Results

Table – Comparison of prediction results of different heuristic algorithms.

Optimization Method	**Turning Point 1**	**Turning Point 2**	**Turning Point 3**
Historical Turning Point	**2008/07/03**	**2011/04/29**	**2016/02/11**
30 days optimized by MPGA	True (2008/06/29–2008/07/29)	True (2011/04/06–2011/05/06)	True (2016/02/10–2016/03/11)
4 weeks optimized by MPGA	True (2008/06/20–2008/07/18)	True (2011/04/15–2011/05/13)	True (2016/01/15–2016/02/12)
30 days optimized by SGA	True (2008/06/26–2008/07/26)	False (2011/09/04–2011/10/04)	False (2016/02/27–2016/03/28)
4 weeks optimized by SGA	False (2008/07/11–2008/08/08)	False (2011/05/20–2011/06/17)	True (2016/02/05–2016/03/04)
30 days optimized by SA	False (2008/06/26–2008/07/26)	False (2011/04/08–2011/05/08)	False (2016/02/19–2016/03/20)
4 weeks optimized by SA	False (2008/06/27–2008/07/25)	False (2011/07/29–2011/08/26)	False (2016/01/15–2016/02/12)
30 days optimized by PSO	False (2008/08/24–2008/09/23)	False (2011/06/17–2011/07/15)	False (2015/10/12–2015/11/11)
4 weeks optimized by PSO	False (2008/07/11–2008/08/08)	False (2011/06/17–2011/07/15)	False (2015/08/21–2015/09/18)

Extensions : MCMC Algorithm

Steps in LPPL Estimation with Metropolis-Hastings

- **Define the Likelihood Function :**

$$P(\text{Data}|\theta) = \exp\left(-\frac{1}{2}\sum_{i=1}^{N}[\ln p(t_i) - f(t_i;\theta)]^2\right)$$

- **Set Priors :** Define prior distributions for θ based on domain knowledge.
- **Proposal Distribution :** Generate candidate parameter sets θ' using a Gaussian random walk.
- **Acceptance Criterion :**

$$r = \frac{P(\text{Data}|\theta')P(\theta')}{P(\text{Data}|\theta)P(\theta)} \quad \text{Accept if } \mathrm{U}(0,1) < \min(1, r).$$

- **Iterate :** Repeat for a large number of samples to approximate the posterior.

Extensions : Firefly Algorithm

Overview

- Inspired by firefly **swarm behavior**, ideal for optimization.
- Effective for **multi-objective, nonlinear problems**.
- Utilizes **global communication** to balance exploration and exploitation.

Key Steps of the Algorithm

At each iteration, fireflies are updated based on :

- **Attraction :** Fireflies move toward brighter ones based on their light intensity.
- **Light Intensity :** Determined by the objective function (RSS for LPPL).
- **Distance Effect :** Attraction decreases with distance due to light absorption.

Extensions : Tabu Search Algorithm

Overview

- Tabu Search is a **metaheuristic algorithm** designed to solve combinatorial and nonlinear optimization problems.
- It explores the solution space by allowing moves that do not always improve the objective function at every step.
- Incorporates **memory structures** (tabu list) to avoid revisiting previously explored solutions.

Main Steps of the Algorithm

- **Initialization** : Start with initial solution and empty tabu list.
- **Move Selection** : Choose the best neighbor, subject to the tabu list constraints.
- **Update** : Accept the new solution if it's better.
- **Tabu List Update** : Add the current solution to the tabu list to prevent revisiting.

Extensions : Reducing Nonlinear Complexity in LPPL

Main Idea

- Reduce interdependence between **phase** (ϕ) and **angular log-frequency** (ω).
- Decrease the number of **nonlinear parameters** in the LPPL equation.

Reformulation Using New Parameters

- LPPL (classic model) :

$$p(t) = A + B(t_c - t)^\alpha + C(t_c - t)^\alpha \cos(\omega \ln(t_c - t) + \phi)$$

- LPPLS (simplified model) :

$$p(t) = A + B(t_c - t)^\alpha + C_1(t_c - t)^\alpha \cos(\omega \ln(t_c - t)) + C_2(t_c - t)^\alpha \sin(\omega \ln(t_c - t))$$

Nelder-Mead Algorithm

Why Nelder-Mead for LPPL ?

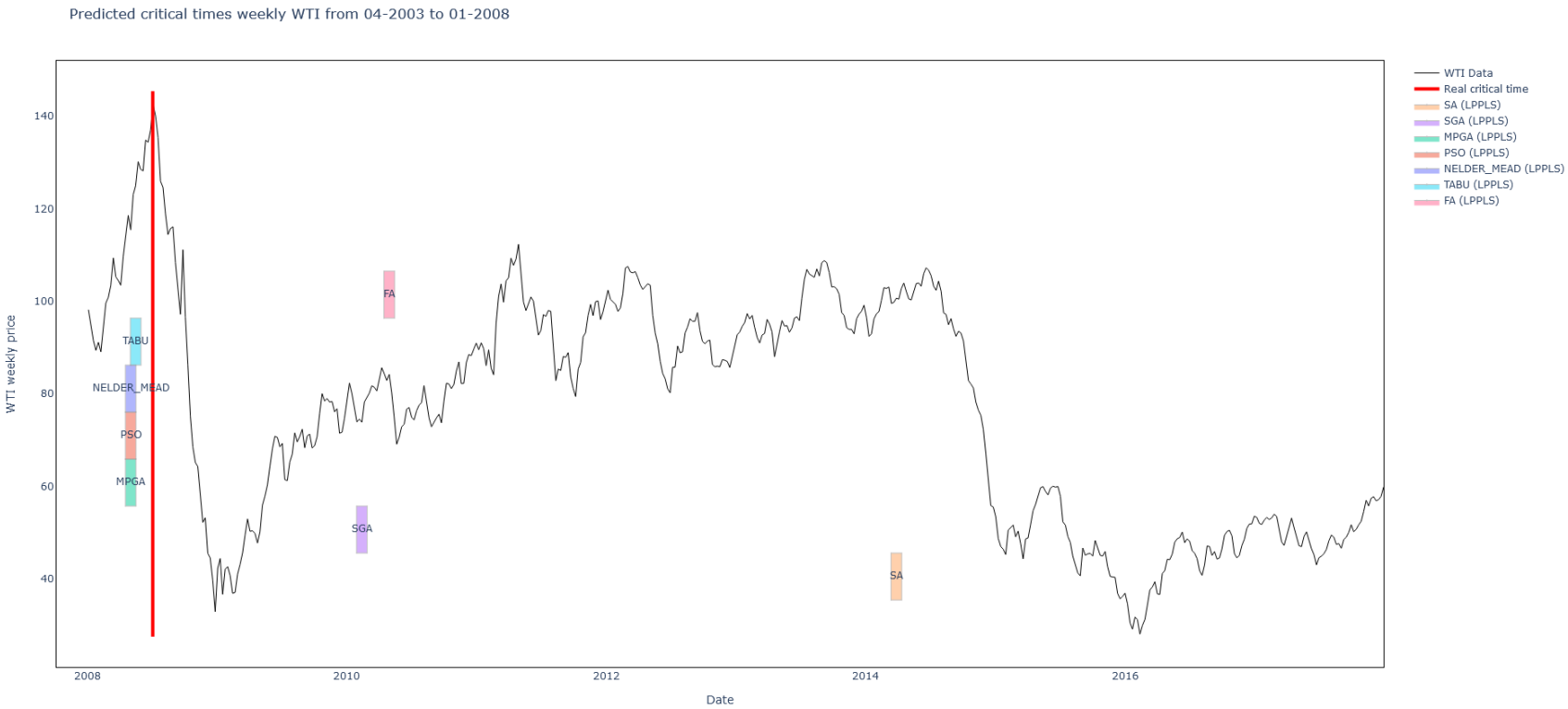
- Suitable for **nonlinear, multidimensional** optimization problems.
- Works well without needing gradient information, ideal for complex LPPL functions.
- Efficient in lower dimensions, (optimizing t_c, α, ω in LPPLS).

Steps of the Algorithm

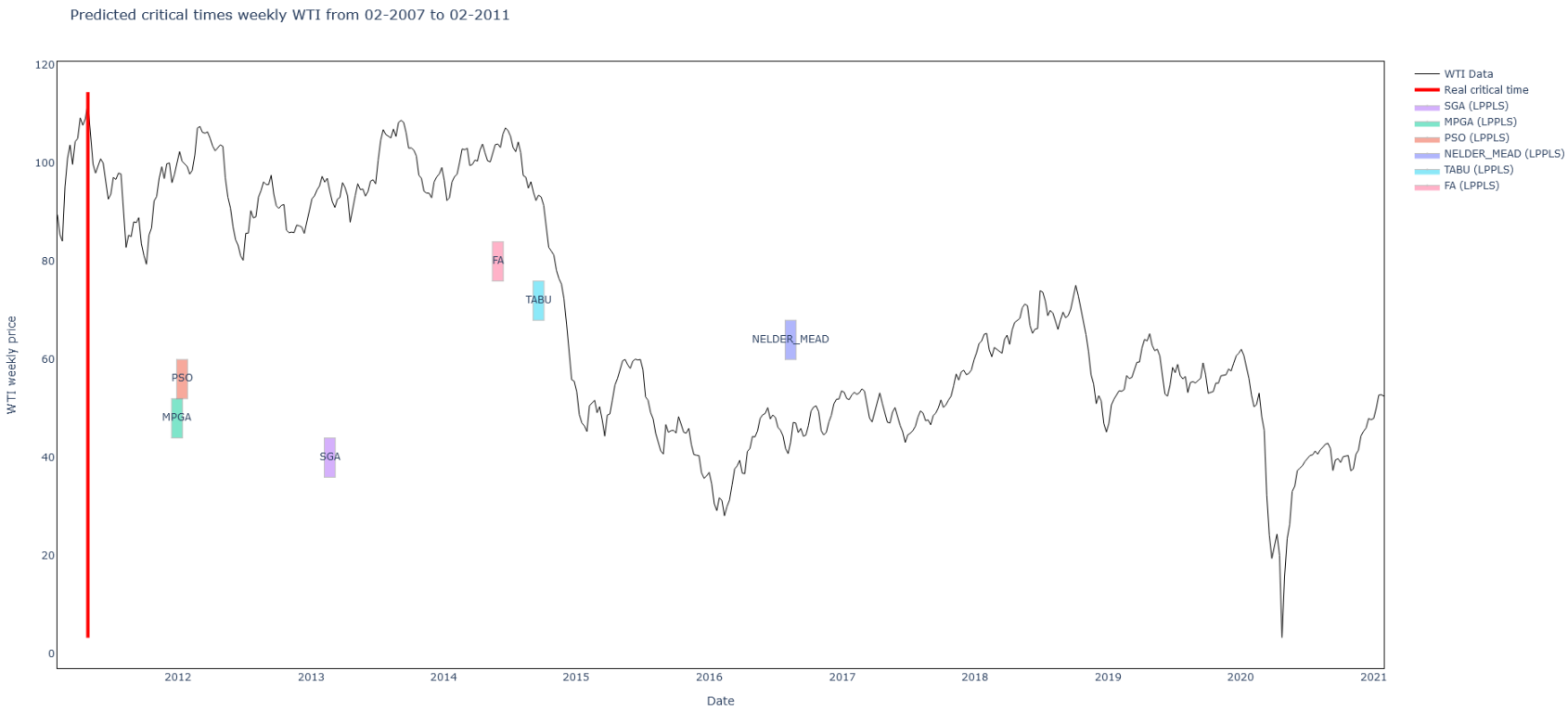
At each iteration, the simplex is modified using :

- **Reflection** : Reflects the worst point across the centroid.
- **Expansion** : Tries to expand along the reflection direction.
- **Contraction** : Shrinks the simplex towards the best point if no improvement.
- **Reduction** : Reduces all points towards the best if the simplex becomes too small.

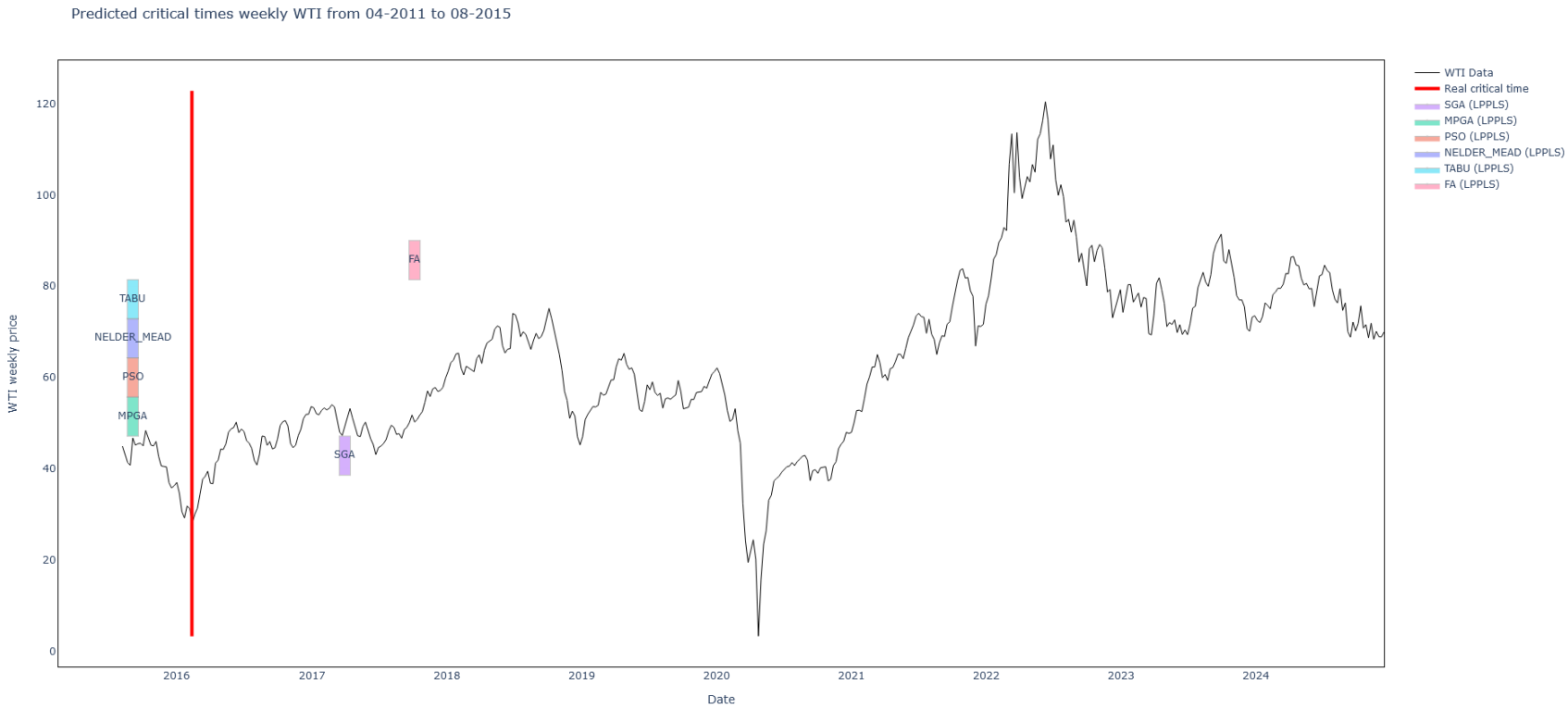
Extensions : Results with LPPLS



Extensions : Results with LPPLS



Extensions : Results with LPPLS



Extensions : Turning point prediction on the SP500

- Graph(s) SP500

Extensions : Turning point prediction on the BTC

- Graph(s) BTC

Extensions : Turning point prediction on the EUR/USD

- Graph(s) EUR/USD

Table – Comparison of prediction results of different heuristic algorithms.

Optimization Method	Turning Point 1	Turning Point 2	Turning Point 3
Historical Turning Point	**2008/07/03**	**2011/04/29**	**2016/02/11**
30 days optimized by MPGA	True (2008/06/29–2008/07/29)	True (2011/04/06–2011/05/06)	True (2016/02/10–2016/03/11)
4 weeks optimized by MPGA	True (2008/06/20–2008/07/18)	True (2011/04/15–2011/05/13)	True (2016/01/15–2016/02/12)
30 days optimized by SGA	True (2008/06/26–2008/07/26)	False (2011/09/04–2011/10/04)	False (2016/02/27–2016/03/28)
4 weeks optimized by SGA	False (2008/07/11–2008/08/08)	False (2011/05/20–2011/06/17)	True (2016/02/05–2016/03/04)
30 days optimized by SA	True (2008/06/26–2008/07/26)	True (2011/04/08–2011/05/08)	False (2016/02/19–2016/03/20)
4 weeks optimized by SA	True (2008/06/27–2008/07/25)	False (2011/07/29–2011/08/26)	True (2016/01/15–2016/02/12)
30 days optimized by PSO	False (2008/08/24–2008/09/23)	False (2011/06/17–2011/07/15)	False (2015/10/12–2015/11/11)
4 weeks optimized by PSO	False (2008/07/11–2008/08/08)	False (2011/06/17–2011/07/15)	False (2015/08/21–2015/09/18)
30 days optimized by Nelder-Mead	True (2008/06/28–2008/07/28)	True (2011/04/10–2011/05/10)	False (2016/02/18–2016/03/19)
4 weeks optimized by Nelder-Mead	True (2008/06/25–2008/07/23)	True (2011/04/13–2011/05/13)	True (2016/01/16–2016/02/13)
30 days optimized by Tabu	False (2008/07/02–2008/08/02)	False (2011/05/08–2011/06/07)	False (2016/02/23–2016/03/24)
4 weeks optimized by Tabu	True (2008/06/29–2008/07/27)	False (2011/05/10–2011/06/09)	True (2016/02/03–2016/03/02)
30 days optimized by MCMC	True (2008/06/30–2008/07/30)	True (2011/04/12–2011/05/12)	True (2016/02/17–2016/03/18)
4 weeks optimized by MCMC	True (2008/06/22–2008/07/20)	True (2011/04/18–2011/05/18)	True (2016/01/14–2016/02/11)

Window parameters finetuning

- 2 years windows

Window parameters finetuning

- 1 years windows

Window parameters finetuning

- 6 years windows

Window parameters finetuning

- 2 month before the real tc

Window parameters finetuning

- 9 month before the real tc

Conclusion

- C'est à chier !

What's next ?

- Insert graph with data of today