

Soutenance Projet de Master

Romain Mencattini

Université de Genève

June 28, 2018

Sommaire

- 1 Introduction
- 2 Algorithme et Implémentation
- 3 Résultats
- 4 Conclusion

Notions de finance

Définition du FOREX¹:

Forex (FX) is the market in which currencies are traded. The forex market is the largest, most liquid market in the world, with average traded values that can be trillions of dollars per day. It includes all of the currencies in the world.

¹Source : <https://www.investopedia.com/terms/f/forex.asp>

Notions de finance

Propriétés du marché :

- Acteurs : les banques commerciales, les entreprises, les institutions financières non-bancaires et les ménages.
- Respecte la règle des trois unités.
- Outils possibles : *spot*, *futur* et *option*.

Notions de *Machine learning*

Définition formelle du *Machine learning*²:

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks in T , as measured by P , improves with experience E .

²Source : "*Machine learning*" by Tom M. Mitchell

Notions de *Machine learning*

Notions importantes :

- Données d'entraînement et de test.
- Apprentissage supervisé.
- Apprentissage non-supervisé.

Motivations

Principales motivations :

- Peu de recherche sur le *machine learning* appliqué au FOREX.
- Prendre en main et améliorer une technique existante.

Sommaire

- 1 Introduction
- 2 Algorithme et Implémentation
- 3 Résultats
- 4 Conclusion

Introduction

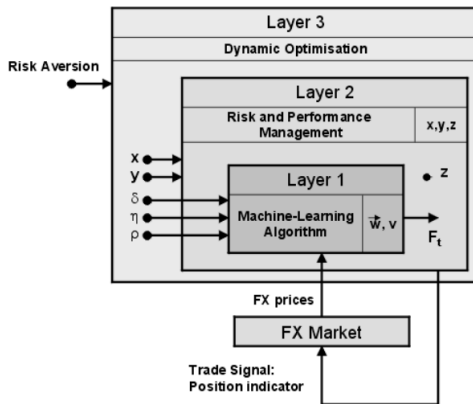


Figure: Image provenant de l'article "An automated FX trading system using adaptive reinforcement learning" de M.A.H. Dempster, V. Leemans

Théorie *Layer 1*

Éléments de la *Layer 1*:

- Le calcul du signal :

$$F_t = \text{sign}\left(\sum_{i=0}^M w_{i,t} r_{t-i} + w_{M+1,t} F_{t-1} + v_t\right)$$

- L'optimisation s'effectue par descente du gradient. Le but est d'optimiser une estimation du *Sharpe ratio* :

$$\hat{S}(t) := \frac{A_t}{B_t}$$

Où $A_t := A_{t-1} + \eta(R_t - A_{t-1})$ et $B_t := B_{t-1} + \eta(R_t^2 - B_{t-1})$

Théorie *Layer 2*

La *Layer 2* est composée de:

- *stop loss*.
- seuillage.



Figure: Effet de la deuxième couche sur un profit cumulé de 3000 points.

Théorie *Layer 3*

Éléments de la *Layer 3* :

- Les fonctions de coûts sont définies comme :

$$\Sigma := \frac{\sum_{i=0}^N (R_i)^2 I(R_i < 0)}{\sum_{i=0}^N (R_i)^2 I(R_i > 0)}$$

$$U(\bar{R}, \Sigma, v) := a \cdot (1 - v) \cdot \bar{R} - v \cdot \Sigma$$

avec $\bar{R} := \frac{P_N}{N}$, v l'aversion au risque et a une constante.

- L'optimisation est donnée par :

$$\max_{\delta, \eta, \rho, x, y} U(\bar{R}; \Sigma : \delta, \eta, \rho, x, y; v)$$

- Afin de rendre le calcul possible, cette formule est préférée :

$$\max_{\delta} \max_{\eta} \max_{\rho} \max_x \max_y U(\bar{R}; \Sigma : \delta, \eta, \rho, x, y; v)$$

Implémentation

Différences avec la théorie :

- Fixer les valeurs initiales.
- Attribuer une valeur lorsqu'une fonction n'est pas définie.
- Compléter les manques.

Problèmes

Les problèmes rencontrés sont :

- ❶ Les poids (w_i) tendent vers l'infini.
- ❷ L'apprentissage est globalement inefficace.
- ❸ L'optimisation des méta-paramètres est bancal.
- ❹ Le programme ralentit au fil de l'exécution.

Solutions

Problème :

Les poids (w_i) tendent vers l'infini.

Solution :

En changeant la descente du gradient par RMSProp :

$$E[g^2]_t = 0.9E[g^2]_{t-1} + 0.1g_t^2$$

avec

$$\theta_{t+1} = \theta_t + \frac{\rho}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

Cela permet d'éviter la divergence.

Solutions

Problème :

L'apprentissage est globalement inefficace.

Solution :

Par l'utilisation du Sharpe ratio comme filtre, on diminue les périodes risquées et les pertes.

Solutions

Problème :

L'optimisation des méta-paramètres est bancale.

Solution :

Fixer les valeurs initiales des méta-paramètres et effectuer une recherche aléatoire normale autour de ces dernières améliorent significativement l'optimisation.

Solutions

Problème :

Le programme ralentit au fil de l'exécution.

Solution :

En évitant de stocker des éléments sur plusieurs itérations lorsque ce n'est pas nécessaire et en utilisant le parallélisme cela résous le problème.

Sommaire

- 1 Introduction
- 2 Algorithme et Implémentation
- 3 Résultats**
- 4 Conclusion

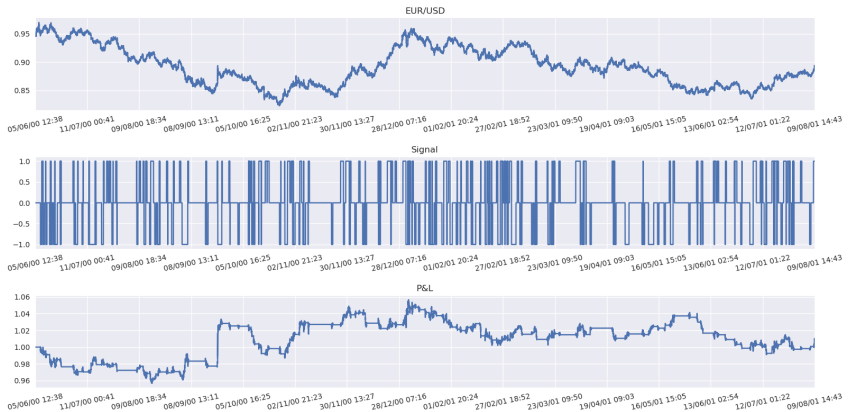


Figure: Cours EUR/USD entre le 01.06.2000 et le 31.09.2001, le signal d'achat prédit par l'algorithme ainsi que le $p\&l$ résultant de ces prédictions. Avec $|w| = 20$, $\delta = 0.001$, $\eta = 0.001$, $\rho = 0.01$, $x = 0.005$, $y = 0.0001$.

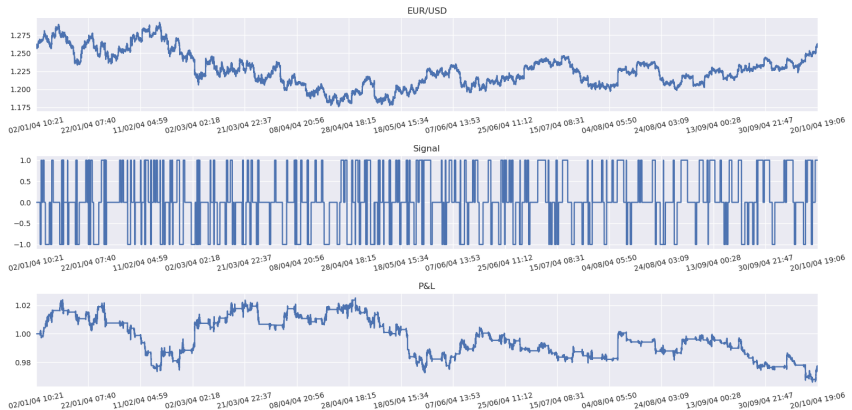


Figure: Cours EUR/USD entre le 01.01.2004 et le 31.10.2004, le signal d'achat prédit par l'algorithme ainsi que le $p\&l$ résultant de ces prédictions. Avec $|w| = 20$, $\delta = 0.001$, $\eta = 0.001$, $\rho = 0.01$, $x = 0.005$, $y = 0.0001$.



Figure: Cours EUR/USD de 2006 à 2010 et le $p\&l$ résultant des prédictions de l'algorithme. Avec $|w| = 20$, $\delta = 0.001$, $\eta = 0.001$, $\rho = 0.01$, $x = 0.005$, $y = 0.0001$.

Vitesse d'exécution

Temps d'exécution des différentes parties du programme :

	Temps (μs)
Calcul F_t	25.5
Descente du gradient	83
Optimisation des paramètres (après 200 itérations)	220
Optimisation des paramètres (après 2 itérations)	2139

Table: Temps d'exécution pour une étape de *backtesting*, avec 2000 éléments d'entraînement et 500 de test, en moyenne sur 1000 runs.

Sommaire

- 1 Introduction
- 2 Algorithme et Implémentation
- 3 Résultats
- 4 Conclusion**

Pistes

Pistes envisageables :

- Ajouter des couches cachées au réseau de neurones.
- Améliorer l'optimisation des méta-paramètres.
- Implémenter une version alternative de la descente du gradient.
- Passer à des méthodes structurelles.

Conclusion

En conclusion :

- Fonctionne localement mais pas globalement.
- Le projet est dans un état fonctionnel à partir d'un article incomplet.
- Cela fournit des pistes de recherches.