Biologie Aujourd'hui **211 (3)**, 233-237 (2017)

https://doi.org/10.1051/jbio/2017029

Disponible en ligne :
www.biologie-journal.org

ARTICLE ORIGINAL

Nextflow : un outil efficace pour l'amélioration de la stabilité numérique des calculs en analyse génomique

Paolo Di Tommaso[1,2], Evan W. Floden[1,2], Cedrik Magis[1,2], Emilio Palumbo[1,2], et Cedric Notredame[1,2,*]

[1] Comparative Bioinformatics Group, Bioinformatics and Genomics Programme, Center for Genomic Regulation (CRG), Dr Aiguader, 88, 08003 Barcelona, Spain
[2] Universitat Pompeu Fabra (UPF), Barcelona, Spain

Reçu le 20 octobre 2017

Résumé– La reproduction des analyses bio-informatiques de routine est difficile en raison d'une combinaison de facteurs difficiles à contrôler. *Nextflow* est un gestionnaire de flux (*workflow manager*) qui utilise la technologie des conteneurs pour assurer un déploiement et une reproductibilité efficace des pipelines d'analyse computationnelle. Les pipelines tiers peuvent être portés dans *Nextflow* avec un recodage minimum. Nous montrons ici à l'aide d'exemples concrets comment la quantification des niveaux d'expression, l'annotation de génomes et la reconstruction de phylogénie peuvent se révéler non reproductibles lorsqu'elles sont réalisées sur des plates-formes UNIX différentes, alors qu'elles deviennent stables lorsqu'elles sont déployées dans *Nextflow*. *Nextflow* est disponible sur www.nextflow.io.

Mots clés : pipelines, *workflow manager*, reproductibilité, *Nextflow*

Abstract – Nextflow, an efficient tool to improve computation numerical stability in genomic analysis. Reproducing routine bioinformatics analysis is challenging owing to a combination of factors hard to control for. *Nextflow* is a flow management framework that uses container technology to insure efficient deployment and reproducibility of computational analysis pipelines. Third party pipelines can be ported into *Nextflow* with minimum re-coding. We used RNA-Seq quantification, genome annotation and phylogeny reconstruction examples to show how two seemingly irreproducible analyzes can be made stable across platforms when ported into *Nextflow*.

Keywords: pipelines, workflow manager, reproducibility, *Nextflow*

La question de plus en plus préoccupante de la non-reproductibilité en biologie expérimentale est la conséquence naturelle du développement de méthodes d'analyse toujours plus complexes (Allison *et al.*, 2016). Malgré l'absence d'un consensus sur une définition commune de la non reproductibilité (Baker, 2016 ; Goodman *et al.*, 2016), il est généralement accepté qu'en biologie computationnelle la principale cause du problème est le manque de pratiques standardisées (*good practices*) lors du déploiement de logiciels et/ou de l'assemblage de banques de données (LeVeque *et al.*, 2012 ; Masca *et al.*, 2015 ; Piccolo & Frampton, 2016). L'absence de standards n'explique cependant pas tout et la reproductibilité limitée des analyses actuelles résulte aussi de facteurs moins apparents tels que l'instabilité numérique causée par des variations techniques au sein des plates-formes de calcul (Garijo *et al.*, 2013). Ce problème est particulièrement sérieux lorsque des environnements de calcul haute performance (*High Performance Computation*, HPC) sont utilisés (Loman & Watson, 2013).

Dans cet article, nous rapportons des cas concrets d'analyses génomiques où des calculs identiques ont donné des résultats différents quand ils ont été réalisés sur des plates-formes de calcul différentes. Cette instabilité est particulièrement inquiétante au moment où la médecine de précision s'apprête à imposer des standards bien plus stricts qu'auparavant quant à la nécessité de réplication analytique. Fort heureusement, des développements techniques récents comme la virtualisation (*i.e.* machines virtuelles) apportent des solutions concrètes à ce problème. Nous montrons ici que la mise en pratique est relativement simple et il ne fait aucun doute que la virtualisation par conteneur jouera un rôle important dans la transition – déjà bien entamée – de la biologie actuelle vers la médecine de précision. À cet effet, nous avons développé un nouvel outil, nommé *Nextflow*.

*Auteur correspondant : cedric.notredame@crg.eu

Influence croissante des gestionnaires de flux sur le calcul scientifique

Nextflow appartient à une catégorie de logiciels couramment désignés sous le terme de *Workflow Manager* (gestionnaire de flux). Ces systèmes permettent de prototyper et de déployer rapidement des pipelines combinant des logiciels complémentaires. Ils sont devenus une partie intégrante des analyses biologiques à haut débit. Les pipelines d'analyse sont les techniques les plus utilisées pour le traitement de données génomiques. Ces pipelines sont en général constitués de l'assemblage informatique de plusieurs logiciels développés indépendamment les uns des autres mais orchestrés ensemble par le pipeline. On parle souvent du pipeline comme d'un script, c'est-à-dire d'un programme écrit dans un langage interprété (Perl, Python, Ruby...), qui permet d'invoquer et de lier des applications tierces (*e.g.* un aligneur multiple et un outil de phylogénie). Le pipeline peut être encapsulé dans un conteneur de type *Docker* afin d'être ensuite déployé par un *Workflow Manager* comme *Nextflow*. L'une des spécificités de *Nextflow* est d'autoriser une encapsulation très fine – outil par outil – dans des conteneurs différents. Il devient ainsi possible de déployer chaque étape du calcul de manière souple et reproductive tout en garantissant une grande stabilité numérique sur des plates-formes UNIX. Ces plates-formes sont de loin les plus couramment utilisées pour les calculs scientifiques liés à l'analyse génomique. Il existe un grand nombre de plates-formes UNIX alternatives. Elles ont en commun de supporter les mêmes langages et les mêmes procédures de gestions de données et de fichiers, mais de nombreux détails d'implémentation peuvent les rendre légèrement différentes (par exemple, les versions des librairies informatiques utilisées pour les calculs de précision).

Le maintien des pipelines génomiques présente des défis importants

Les systèmes de gestion de flux *in silico* sont récemment devenus partie intégrante de l'analyse biologique. En génomique, les pipelines les plus simples, tels que *Kallisto* et *Sleuth* (Bray *et al.*, 2016), impliquent la combinaison d'une méthode de quantification ARN-seq avec un module d'analyse de l'expression différentielle. La complexité des pipelines croît rapidement lorsque leur fonction est de couvrir tous les aspects d'une tâche d'analyse spécifique. Par exemple, le pipeline *Sanger Companion* (Steinbiss *et al.*, 2016) regroupe 39 outils logiciels et bibliothèques indépendants au sein d'une suite d'annotations génomiques destinées à l'annotation des eukaryotes unicellulaires. Manipuler un si grand nombre de logiciels est un défi à part entière. En effet, le développement indépendant de chaque élément ne permet pas de présager de leur compatibilité et il peut très bien arriver que des dépendances spécifiques rendent leur coexistence au sein d'un même système informatique impossible, ou tout du moins compliquée à mettre en place pour un non-spécialiste. Ce problème est encore aggravé par l'absence de coordination entre les mises à jour et la nécessité de maintenir reproductibles les résultats initialement publiés – souvent les seuls validés scientifiquement. Le déploiement à haut débit de pipelines complexes peut également être problématique, en raison du très grand nombre de fichiers intermédiaires produits par les étapes successives d'un pipeline. À cette échelle, les fluctuations matérielles probables combinées à une mauvaise gestion des erreurs peuvent entraîner une instabilité de calcul.

Nextflow répond au problème du déploiement reproductible de calculs génomiques

Nextflow (Di Tommaso *et al.*, 2017) a été spécifiquement conçu pour régler les problèmes suivants (par ordre d'importance) :
- l'instabilité numérique ;
- le déploiement parallèle efficace de calculs ;
- la tolérance aux erreurs (*i.e.* la capacité de relancer un calcul partiellement réalisé) ;
- la provenance de l'exécution et la traçabilité de la maintenance (*i.e.* la possibilité de lancer des versions anciennes d'un pipeline).

Il s'agit d'un langage de domaine spécifique (DSL) qui permet un prototypage rapide des pipelines ainsi que l'adaptation rapide de pipelines existants écrits dans n'importe quel langage interprété. Une comparaison qualitative entre *Nextflow* et plusieurs outils similaires (Di Tommaso *et al.*, 2017) illustre bien la combinaison unique de ses caractéristiques, notamment en comparaison de *Bpipe*, dont le paradigme de calcul est relativement similaire. L'inconvénient le plus notable de *Bpipe* est son manque de support pour la conteneurisation multi-échelle, l'une des fonctionnalités désormais supportée par la toute dernière génération de gestionnaires de flux de travail, y compris *Nextflow*. Cette manière d'encapsuler les calculs offre la possibilité d'orchestrer des outils installés, soit dans le même conteneur, soit dans des conteneurs différents. Les conteneurs séparés présentent un avantage considérable pour ce qui est de la maintenance. En effet, une fois installés dans un conteneur privé, un logiciel peut être mis à jour sans aucun risque d'interférence entre les librairies dont il dépend et celles dont dépendent d'autres logiciels installés dans d'autres conteneurs. Cette individualisation est essentielle à la stabilité numérique des analyses, en particulier quand des analyses effectuées sur de longues périodes de temps doivent être comparées. Ce mode de conteneurisation permet donc de regrouper des pipelines entiers, des sous-composantes et des outils individuels dans leurs propres conteneurs.

La production de conteneurs de calculs peut être standardisée

Les conteneurs peuvent être produits *ad hoc* par les auteurs ou en suivant les standards récemment proposés, *BioBoxes* (Belmann *et al.*, 2015), *Bioshadock*

(Moreews *et al.*, 2015) et *AlgoRun* (Hosny *et al.*, 2016). Une autre spécificité clef de *Nextflow* est son intégration complète avec des référentiels logiciels tels que *GitHub* et *BitBucket*, ainsi que la prise en charge native des nuages de calcul les plus communs (Amazon, Microsoft). L'intégration de ces ressources publiques est un facteur clef de la reproductibilité informatique. En particulier, l'impact spécifique de *GitHub* a récemment été mis en évidence comme l'un des moteurs de l'effort de partage de données. En pratique, l'intégration avec *GitHub* permet d'assurer le déploiement de la version correcte de n'importe quel pipeline existant. Par ailleurs, la conteneurisation du pipeline garantit une stabilité numérique. Enfin, le déploiement sur un nuage de calcul offre une mise en échelle quasi illimitée. Il est intéressant de noter que cette procédure permet d'associer à n'importe quel résultat – un graphe, un tableau, un chiffre clef – à une seule ligne de commande qui peut être référencée, mise à jour, reproduite ou améliorée à la demande. Dans la dernière section de ce manuscrit, nous fournissons trois exemples concrets de cette procédure.

Nextflow utilise un paradigme de calcul *top–down* similaire au pipe de UNIX et différent de celui de *Makefile*

Nextflow utilise un modèle de programmation réactive fonctionnelle (FRP) dans lequel chaque opération est isolée dans son propre contexte d'exécution. Les sorties sont transmises à d'autres opérations *via* des canaux de communication dédiés dans un processus similaire à celui des pipes en UNIX. Cette approche fait de la parallélisation une conséquence implicite de la manière dont les entrées/sorties de chaque processus sont déclarées et elle évite donc aux utilisateurs la nécessité de mettre en œuvre une stratégie de parallélisation explicite. Un autre avantage de *Nextflow* est son utilisation du paradigme de programmation en flux de données. Dans ce paradigme, le déclenchement des tâches s'effectue automatiquement aussitôt les données reçues par leurs canaux d'entrée (*e.g.* déclenchement automatique d'une opération quand un fichier est créé dans un dossier). Ce modèle est supérieur aux solutions alternatives basées sur une approche ressemblant à un *Makefile*, comme *Snakemake* (Koster & Rahmann, 2012), dans lequel le calcul implique la pré-estimation de toutes les dépendances, en commençant à partir des résultats attendus et en remontant jusqu'aux données brutes. Cette procédure nécessite un graphe orienté acyclique (*Direct Acyclic Graph*, DAG), dont l'exigence de stockage est un facteur limitant pour les très grands calculs. *Nextflow* n'a pas besoin du pré-calcul d'un DAG car son modèle de traitement – qui commence par les données brutes – suit le flux naturel d'analyse des données. Ainsi, le graphe que traverse *Nextflow* est simplement incidental et n'a donc pas besoin d'être pré-calculé, ni même stocké. L'utilisation de canaux de communication entre les tâches contribue également à la robustesse computationnelle de *Nextflow*, en particulier en comparaison de *Snakemake*, dont la séquence d'exécutions de tâche est définie par des règles et des modèles définis sur les noms de fichiers entrée/sortie. Ces dépendances rendent difficile la gestion des fichiers de sortie multiples/variables et nécessitent souvent la mise en œuvre de procédures de gestion des sorties de bas niveau pour gérer les différentes étapes d'un pipeline.

Nextflow est un outil de codage différent de *Galaxy*

Nextflow peut gérer n'importe quelle structure de données sans limitation liée à la nature des fichiers. *Nextflow*, comme la dernière génération de gestionnaires de flux de travail (Perkel, 2016), est un outil explicitement conçu pour les bioinformaticiens et pour un prototypage par ligne de commande (*i.e.* sans interface graphique). Cela le différencie clairement de *Galaxy*, l'un des systèmes de flux de travail les plus populaires (Goecks *et al.*, 2010). *Galaxy* peut gérer le problème de la stabilité numérique grâce à un gestionnaire de paquets personnalisé appelé *Tool Shed* (Blankenberg *et al.*, 2014). Son interface graphique offre un support très puissant pour la mise en œuvre d'un pipeline *de novo* par des non spécialistes mais elle impose également une lourde charge de développement dans le cas de pipelines tiers existants, en particulier lorsqu'il s'agit de combinaisons complexes d'outils. Cette exigence de ré-implémentation est commune à des outils autres que *Galaxy* comme *Toil* (Vivian, 2016).

Nextflow offre une grande répétabilité de calcul

Afin de montrer l'effet concret de la variabilité environnementale sur la stabilité numérique, nous avons utilisé le *Sanger Companion pipeline* (Steinbiss *et al.*, 2016) pour effectuer l'annotation du génome *Leishmania infantum* (Di Tommaso *et al.*, 2017). Bien que ce génome eucaryote compact soit une cible relativement facile pour une telle analyse, nos résultats ont indiqué des variations sur différentes platformes UNIX (Figure 1A, B). Cette instabilité contraste avec le comportement déterministe mesuré sur chaque plate-forme individuelle (*i.e.* les résultats sont reproductibles sur une plate-forme donnée, mais pas d'une plate-forme à l'autre). Dans la mesure où le *Sanger* a fait le choix d'implémenter le pipeline *Companion* à l'aide de *Nextflow*, nous avons pu confirmer la stabilité des annotations lors du déploiement d'une version conteneurisée de ce même pipeline sur trois différents systèmes d'exploitation de type UNIX. L'annotation de génome n'est pas le seul type d'analyse sensible à la plate-forme de calcul. Nous avons identifié des problèmes similaires lors de l'utilisation de l'outil de quantification d'expression RNASeq de *Kallisto* combiné avec le paquet d'expression différentielle de *Sleuth* (Bray *et al.*, 2016). Dans ce cas, des variations dans l'identification des gènes différentiellement exprimés apparaissent lors de l'exécution du pipeline sur deux systèmes différents (Figure 1C).

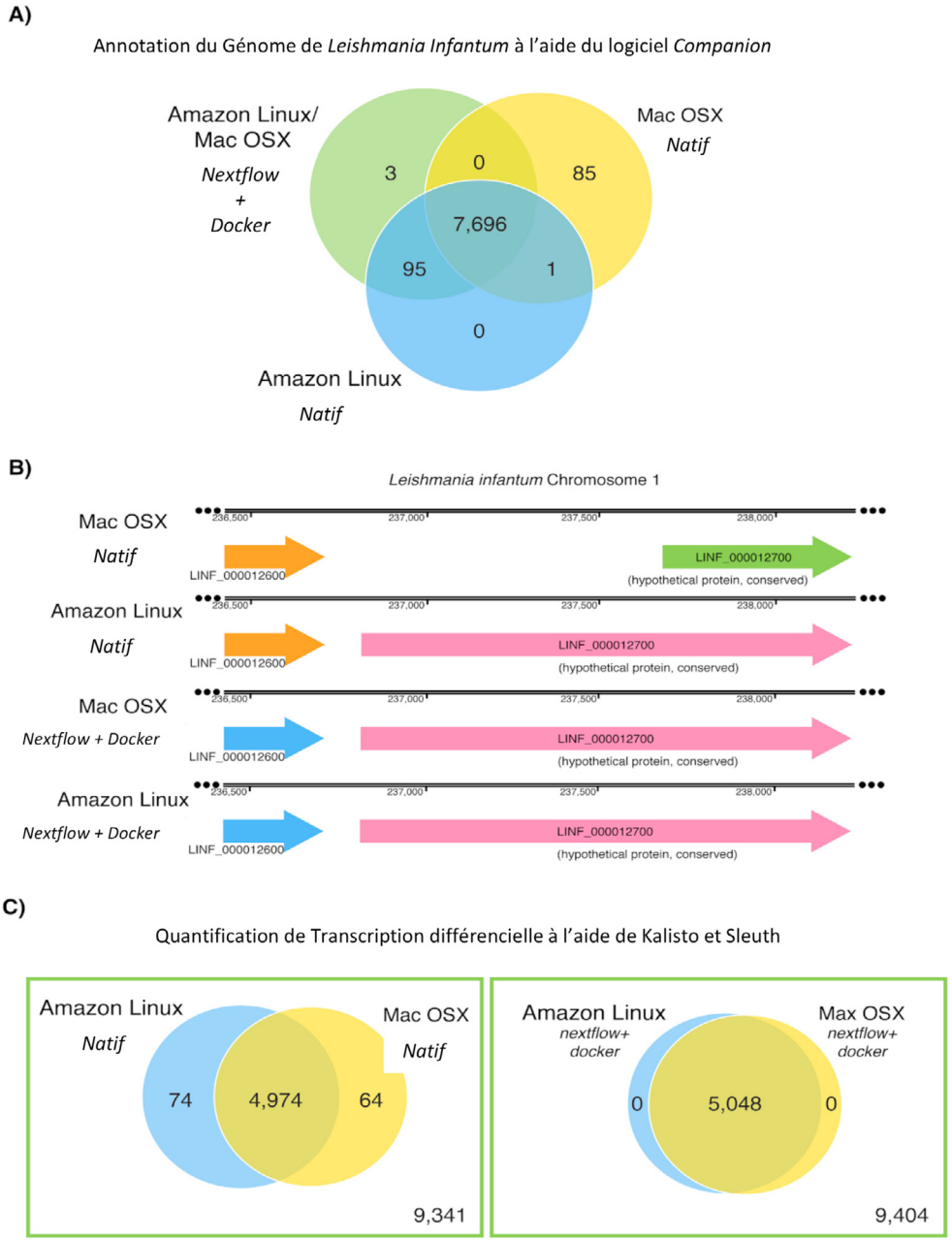

Figure 1. *Nextflow* produit des résultats stables sur des plates-formes UNIX différentes. (A) L'annotation du génome de *Leishmania infantum* JPCM5 a été prédite à l'aide d'une version native et dockerisée (Debian Linux) du pipeline d'annotation eucaryote *Companion*. Les versions natives et dockerisées étaient exécutées sur les plates-formes Mac OSX et Amazon Linux. Le diagramme de Venn montre l'existence de différences minimes mais significatives lors de la comparaison des coordonnées génomiques des gènes codants prédits et des ARN non codants. (B) Certaines de ces disparités incluent des gènes entiers. Au contraire, la version incluse dans un conteneur *docker* donne le même résultat sur toutes les plates-formes. (C) Une comparaison similaire a été réalisée sur un pipeline *Kallisto*/*Sleuth* lors de la recherche de gènes différentiellement exprimés (valeur $q < 0{,}01$) dans une expérience ARN-seq (fibroblastes pulmonaires humains). L'analyse réalisée sur des plates-formes Mac OSX et Amazon Linux montre des disparités minimes mais significatives qui disparaissent lorsque la version dockérisée est déployée.

Toutefois, aucune différence de ce type n'a été observée lors de l'exécution d'une version conteneurisée du même pipeline déployée à l'aide de *Nextflow*. Enfin, un effet identique a été observé lors de l'estimation d'arbres phylogénétiques de maximum de vraisemblance avec *RaxML* (Stamatakis, 2006). Ces variations ont été efficacement contrôlées lors du déploiement de la version dockerisée du même pipeline. Il est à noter que toutes ces expériences de calcul sont disponibles sur *GitHub* (pipelines) et *Zenodo* (données/résultats) et sont donc entièrement reproductibles sous leur forme dockérisée (Di Tommaso *et al.*, 2017).

Conclusion

Nextflow offre une solution simple mais puissante au problème de l'instabilité numérique lors du déploiement de pipelines d'analyses génomiques. Nous montrons ici que cette instabilité, très fréquente, affecte la plupart des types de modélisation réalisés *in silico*. Son impact global sur les analyses finales peut sembler modeste, mais en l'absence de solution standard, ces fluctuations génèrent une instabilité actuellement insurmontable car un contrôle attentif des versions de base de données et de logiciels n'est pas suffisant. Le manque de stabilité numérique peut avoir de graves conséquences à tous les niveaux de l'analyse. Ainsi, lors du traitement de données expérimentales, l'instabilité peut compromettre les vérifications et les mises à jour de résultats déjà établis. Par ailleurs, dans un environnement de production de type médecine personnalisée, l'instabilité peut entraîner des variations arbitraires de traitement aux conséquences potentiellement dramatiques. La communauté d'utilisateurs *Nextflow*, qui connaît une croissance rapide (Di Tommaso, 2016 ; Kurs *et al.*, 2016), illustre le besoin pressant de disposer de cadres de calcul plus robustes. À une époque où la technologie évolue à un rythme effréné, *Nextflow* offre une solution mature à l'un des rares problèmes techniques susceptibles d'être durables tant dans le cadre académique que clinique : le contrôle de la stabilité numérique (Byron *et al.*, 2016).

Références

Allison, D.B., Brown, A.W., George, B.J., Kaiser, K.A. (2016). Reproducibility: A tragedy of errors. *Nature*, 530, 27-29.

Baker, M. (2016). Muddled meanings hamper efforts to fix reproducibility crisis. *Nature News*. DOI: 10.1038/nature.2016.20076.

Belmann, P., Dröge, J., Bremges, A., McHardy, A.C., Sczyrba, A., Barton, M.D. (2015). Bioboxes: standardised containers for interchangeable bioinformatics software. *Gigascience*, 4, 47.

Blankenberg, D., Von Kuster, G., Bouvier, E., Baker, D., Afgan, E., Stoler, N; Galaxy Team, Taylor J., Nekrutenko, A. (2014). Dissemination of scientific software with Galaxy ToolShed. *Genome Biol*, 15, 403.

Bray, N. L., Pimen tel, H., Melsted, P., Pachter, L. (2016). Near-optimal probabilistic RNA-seq quantification. *Nat Biotechnol*, 34, 525-527.

Byron, S.A., Van Keuren-Jensen, K.R., Engelthaler, D.M., Carpten, J.D., Craig, D.W. (2016). Translating RNA sequencing into clinical diagnostics: opportunities and challenges. *Nat Rev Genet*, 17, 257-271.

Di Tommaso, P. (2016). A curated list of Nextflow pipelines. Available at: https://github.com/nextflow-io/awesome-nextflow/ (Accessed: 18th May 2016).

Di Tommaso, P., Chatzou, M., Floden, E.W., Prieto Barja, P., Palumbo, E., Notredame, C. (2017). Nextflow enables reproducible computational workflows. *Nat Biotech*, 35, 316-319.

Garijo, D., Kinnings, S., Xie, L., Xie, L., Zhang, Y., Bourne, P.E., Gil, Y. (2013). Quantifying reproducibility in computational biology: the case of the tuberculosis drugome. *PLoS One*, 8, e80278.

Goecks, J., Nekrutenko, A., Taylor, J. (2010). Galaxy Team. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol*, 11, R86.

Goodman, S. N., Fanelli, D., Ioannidis, J.P.A. (2016). What does research reproducibility mean? *Sci Transl Med*, 8, 341ps12.

Hosny, A., Vera-Licona, P., Laubenbacher, R., Favre, T. (2016). AlgoRun: a Docker-based packaging system for platform-agnostic implemented algorithms. *Bioinformatics*, 32, 2396-2398.

Koster, J., Rahmann, S. (2012). Snakemake-a scalable bioinformatics workflow engine. *Bioinformatics*, 28, 2520-2522.

Kurs, J.P., Simi, M., Campagne, F. (2016). NextflowWorkbench: Reproducible and reusable workflows for beginners and experts. *bioRxiv*, 041236. DOI: 10.1101/041236.

LeVeque, R.J., Mitchell, I.M., Stodden, V. (2012). Reproducible research for scientific computing: Tools and strategies for changing the culture. *Comput Sci Eng*, 14, 13-17.

Loman, N., Watson, M. (2013). So you want to be a computational biologist? *Nat Biotechnol*, 31, 996-998.

Masca, N.G., Hensor, E.M., Cornelius, V.R., Buffa, F.M., Marriott, H.M., Eales, J.M., Messenger, M.P., Anderson, A.E., Boot, C., Bunce, C., Goldin, R.D, Harris, J., Hinchliffe, R.F., Junaid, H., Kingston, S., Martin-Ruiz, C., Nelson, C.P., Peacock, J., Seed, P.T., Shinkins, B., Staples, K.J., Toombs, J., Wright, A.K., Teare, M.D. (2015). RIPOSTE: a framework for improving the design and analysis of laboratory-based research. *Elife*, 4. DOI: 10.7554/eLife.05519.

Moreews, F., Sallou, O., Ménager, H., Le Bras, Y., Monjeaud, C., Blanchet, C., Collin, O. (2015). BioShaDock: a community driven bioinformatics shared Docker-based tools registry. *F1000Res*, 4, 1443.

Perkel, J. (2016). Democratic databases: science on GitHub. *Nature News*, 538, 127.

Piccolo, S.R., Frampton, M.B. (2016). Tools and techniques for computational reproducibility. *Gigascience*, 5, 30.

Stamatakis, A. (2006). RAxML-VI-HPC: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. *Bioinformatics*, 22, 2688-2690.

Steinbiss, S., Silva-Franco, F., Brunk, B., Foth, B., Hertz-Fowler, C., Berriman, M., Otto, T.D. (2016). Companion: a web server for annotation and analysis of parasite genomes. *Nucleic Acids Res*, 44, W29-34.

Vivian, J. (2016). Rapid and efficient analysis of 20, 000 RNA-seq samples with Toil. *bioRxiv*, 062497. DOI: 10.1101/062497.

Citation de l'article : Tommaso, P.D., Floden, E.W., Magis, C., Palumbo, E., et Notredame, C. (2017). *Nextflow* : un outil efficace pour l'amélioration de la stabilité numérique des calculs en analyse génomique. *Biologie Aujourd'hui*, **211**, 233-237