Retour d’expérience sur la migration d’un modèle de projection des flux de trésorerie dans un langage open source.

Florian Thomas-Laglayse – Partner Re

Arthur Maillart - Detralytics

Plan de l’atelier

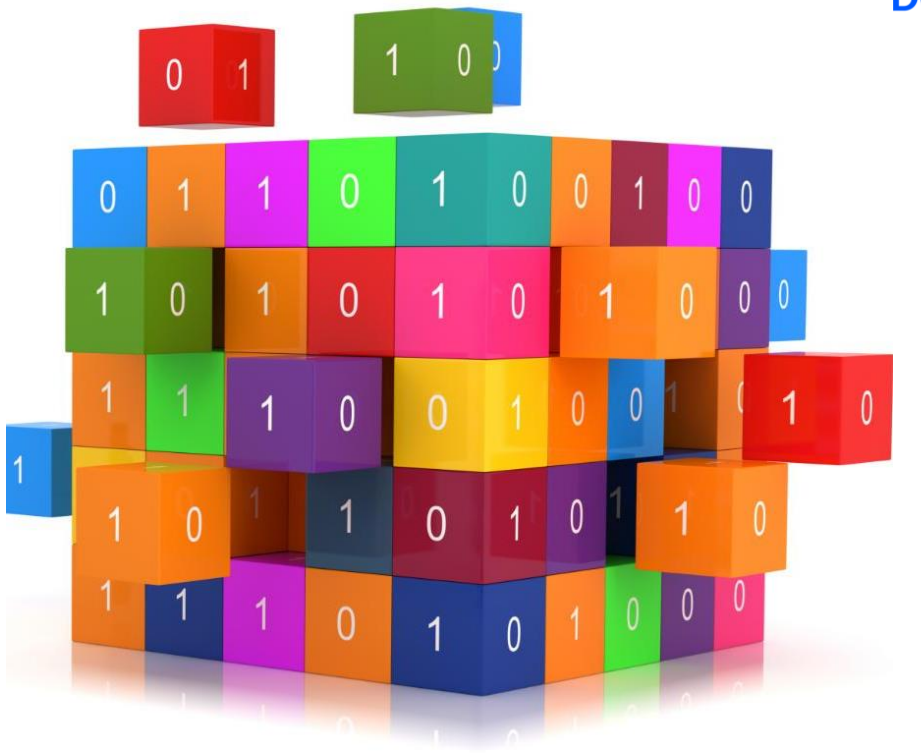
Contexte

Processus de production

Axes d'analyse

Bilan

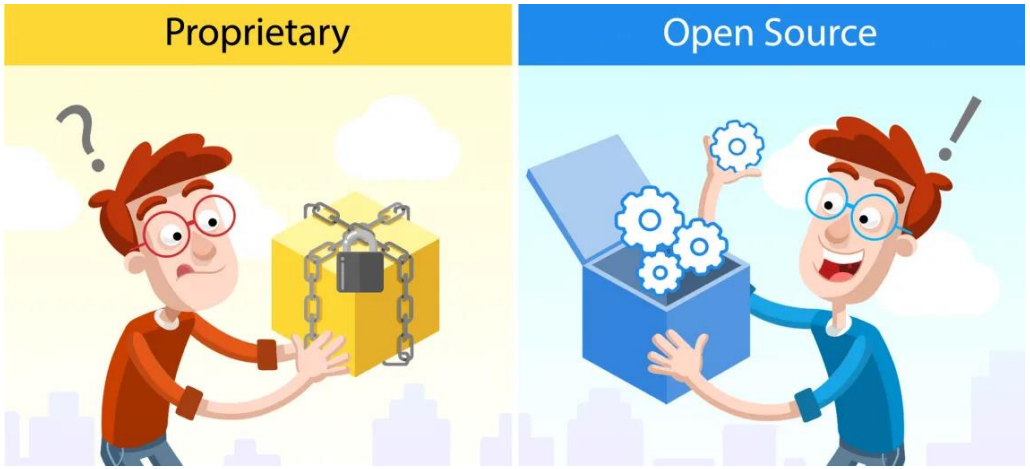
Perspectives



Contexte marché

Constats:

- Une **dépendance** forte à l'éditeur et un service parfois en rupture avec les besoins de l'entreprise
- Des **coûts** importants liés au logiciel, à sa maintenance et au système informatique autour de la solution
- Des **temps de calculs** pouvant être longs
- Un **audit** et une **réplicabilité** des calculs difficiles (black / glass box)
- Des **processus de production lourds** qui empiètent sur le temps des travaux d'analyse
- Des automatisations difficiles à mettre en place avec des **logiciels trop rigides**
- Une gestion des **ressources humaines difficiles** avec des expertises qui disparaissent
- Une **dette technique** importante avec une maîtrise partielle du code et difficile à maintenir
- Un **risque** de ne pas pouvoir répondre aux nouvelles exigences règlementaires

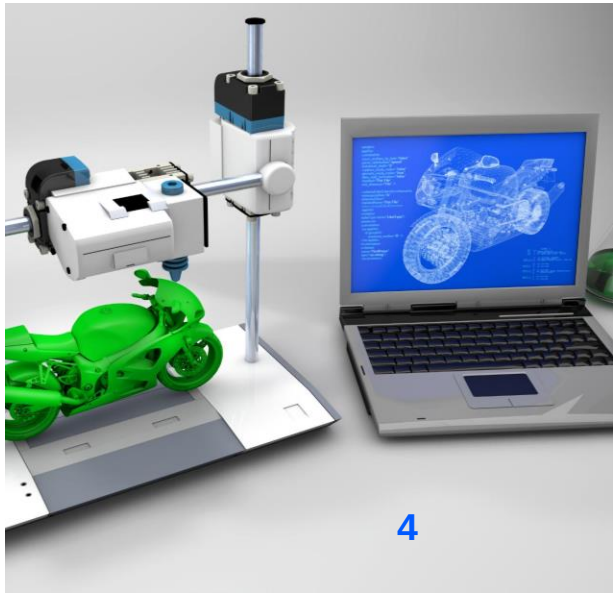


Contexte interne

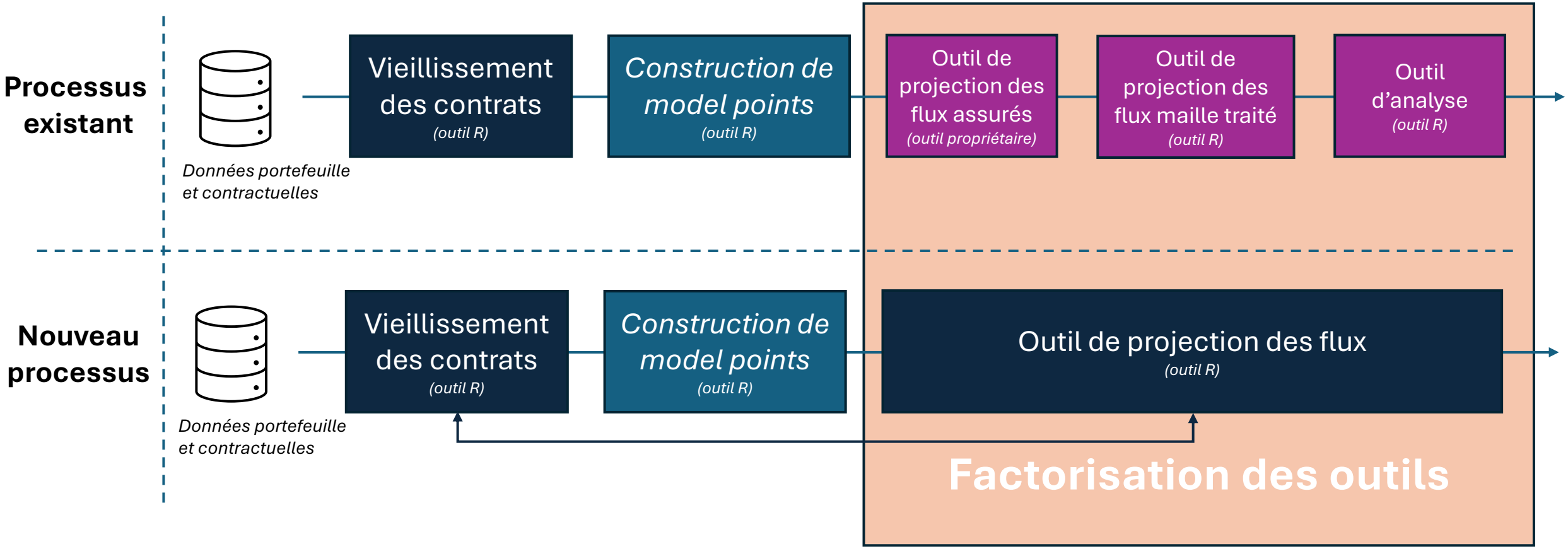
- **Evolutions du métier**
  - Innovation des produits
  - Mutations réglementaires
  - Des délais raccourcis pour des demandes accrues (plus vite, plus fiable, plus complet)
- **Outil actuel**
  - Des coûts financiers et matériels liés à l'exécution de l'outil et des services annexes (licence, machine virtuelle, cloud, processus opérationnel)
  - Migration forcée et onéreuse
  - Une modélisation « rigide » par rapport aux besoins métiers et prudentiels
  - Etudes de migration chez de multiples éditeurs de logiciel

- **Une gouvernance des outils internes centrée sur le langage R :**
  - Une infrastructure IT dédiée (gitlab, plateforme cloud)
  - Des procédures de contrôle s'appuyant sur gitlab (Access Control Policy, Change management)
  - Un « centre de compétences » disposant de profils experts en développement R
  - Un besoin d'aligner l'approche actuarielle sur l'ensemble de la chaine de valeur (tarification, gestion des données, valorisation, demandes internes)

Modernisation nécessaire des outils et de l'environnement de production



Processus de production



Dispositif du projet

Caractéristique	Projet R	Projet Progiciel
Gestion projet	Métier	Equipe dédiée
Équipe de développement	Intégrée avec le métier	Centralisée / Intégrée
Nombre d'internes (ETP*)	1	1
Nombre de consultants (ETP)	1	4
Durée du projet (en j)	70	> +70
Coût total de la migration	<= €150k	> €500k
Durée ROI	3 trimestres	Quid

Un projet à taille humaine avec un fort effet de levier opérationnel et financier

Dimensions de l’analyse

Temps de calcul

Charge opérationnelle

Coûts des solutions

Gouvernance

Dette technologique

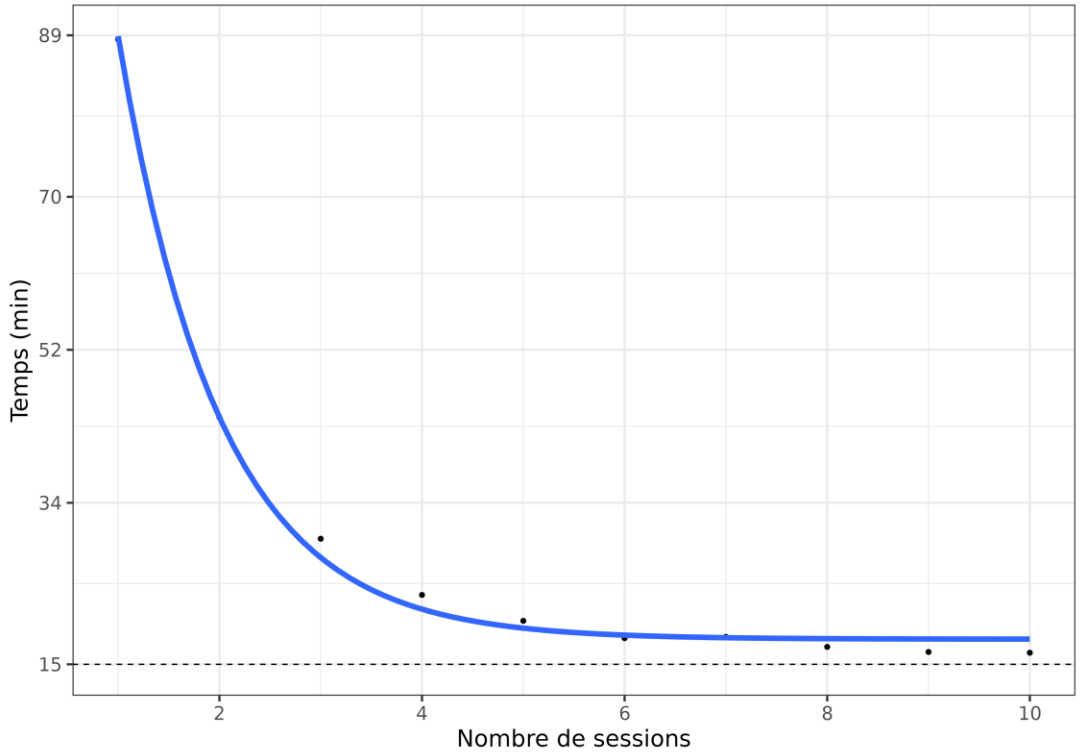
Ressources humaines

Empreinte carbone

Temps de calcul

Les langages tels que R et Python disposent en général d'outils permettant d'optimiser le temps de calcul. Pour R, nous avons mis en place les techniques suivantes:

- **Vectorisation :** R est optimisé pour les opérations vectorielles. Nous avons utilisé des fonctions vectorisées (qui opèrent sur des vecteurs / matrices sans boucles for).
- **Utilisation de packages optimisés :** Des packages comme `data.table`, `dplyr` ou `purrr` offrent des implémentations rapides et performantes pour manipuler des données.
- **Profilage avec `profvis` / `Rprof` :** Utiliser ces outils pour identifier les parties du code qui consomment le plus de temps / mémoire.
- **Réduction des appels répétés :** Nous avons fait un important travail de rationalisation et de structuration du code pour éviter les calculs redondants.
- **Parallelisation :** Plusieurs packages R dont `future,` permettent de paralléliser les calculs efficacement.
- **Utilisation de structures adaptées :** Nous avons privilégié l'utilisation de matrices sur des portions de codes nécessitant une vitesse de calcul plus élevée.
- **Rcpp et RcppArmadillo :** Pour de rares fonctions, trop complexes pour être vectorisées, nous avons écrit ces dernières en C++ grâce à `Rcpp`.



Temps de calcul

Processus de production complet clôture multi normes

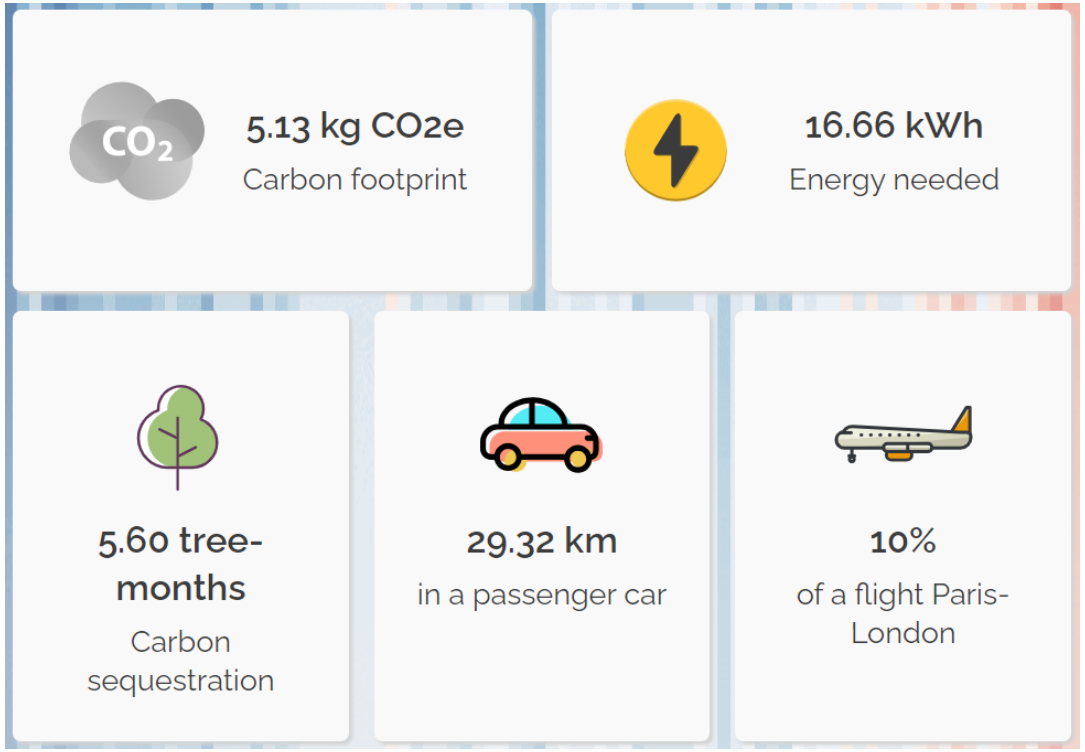
		CPU	RAM	Temps pour lancer tous les BE (min)	Economies (% temps)
Progiciel	Option 1	1 ou plusieurs best estimates (1024 cœurs)	7697Gb	320	
Outil R	Option 1	1 best estimate (6 coeurs)	40Gb	225	-30%
Outil R	Option 2	4 best estimates (4 x 6 coeurs)	160Gb	60	-81%
Outil R	Option 3	Azure batches (6 cœurs par batch)	40Gb par batch	15	-95%

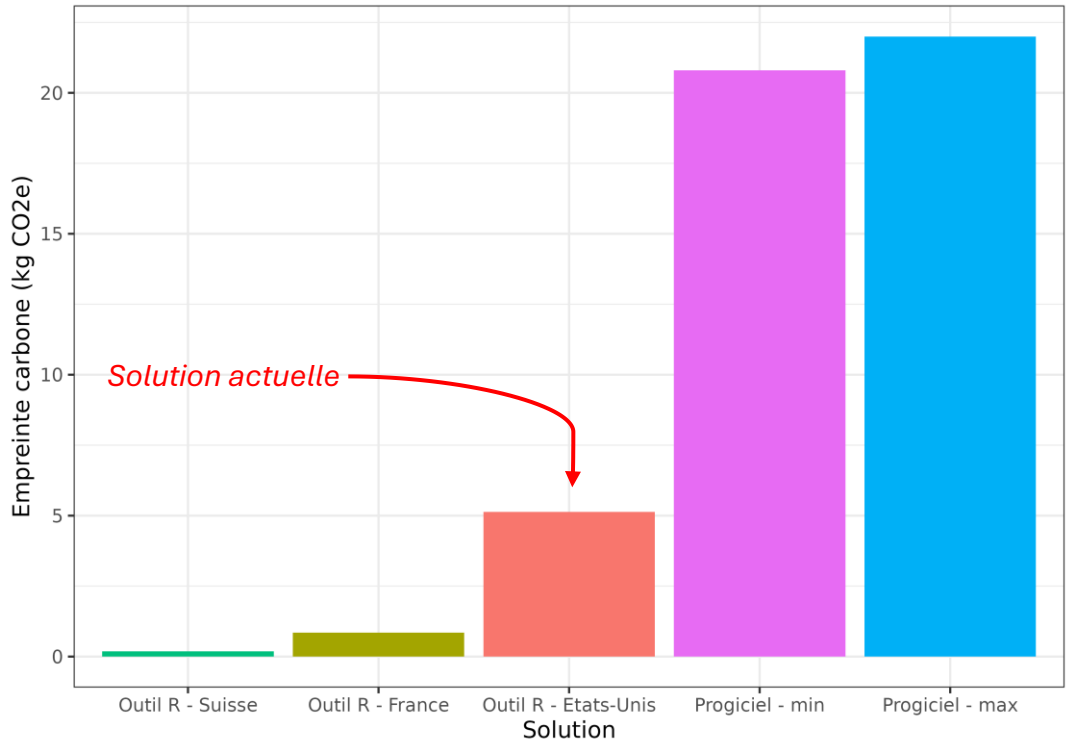
Runs trimestriels

		CPU	RAM	Temps pour lancer tous les BE (min)	Economies (% temps)
Progiciel	Option 1	1 ou plusieurs best estimates (1024 cœurs)	7697Gb	770	
Outil R	Option 1	1 best estimate (6 coeurs)	40Gb	135	-58%
Outil R	Option 2	4 best estimates (4 x 6 coeurs)	160Gb	45	-86%
Outil R	Option 3	Azure batches (6 cœurs par batch)	40Gb par batch	15	-95%

Analyse de mouvements

Empreinte carbone





Source: https://www.green-algorithms.org/

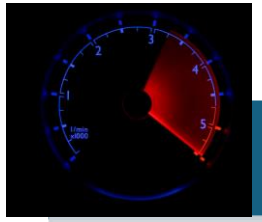
Charge opérationnelle

Projet

- Expression du besoin technique
- Lien direct avec le développeur
- Fluidité de la boucle métier/développeur
- Documentation intégrée
- Une implémentation sur-mesure pour un temps optimisé

Maintenance

- Remontée et suivi des sujets facilités par l'interface gitlab
- Réactivité d'implementation des solutions
- Gouvernance facilitée
- Traçabilité et auditabilité

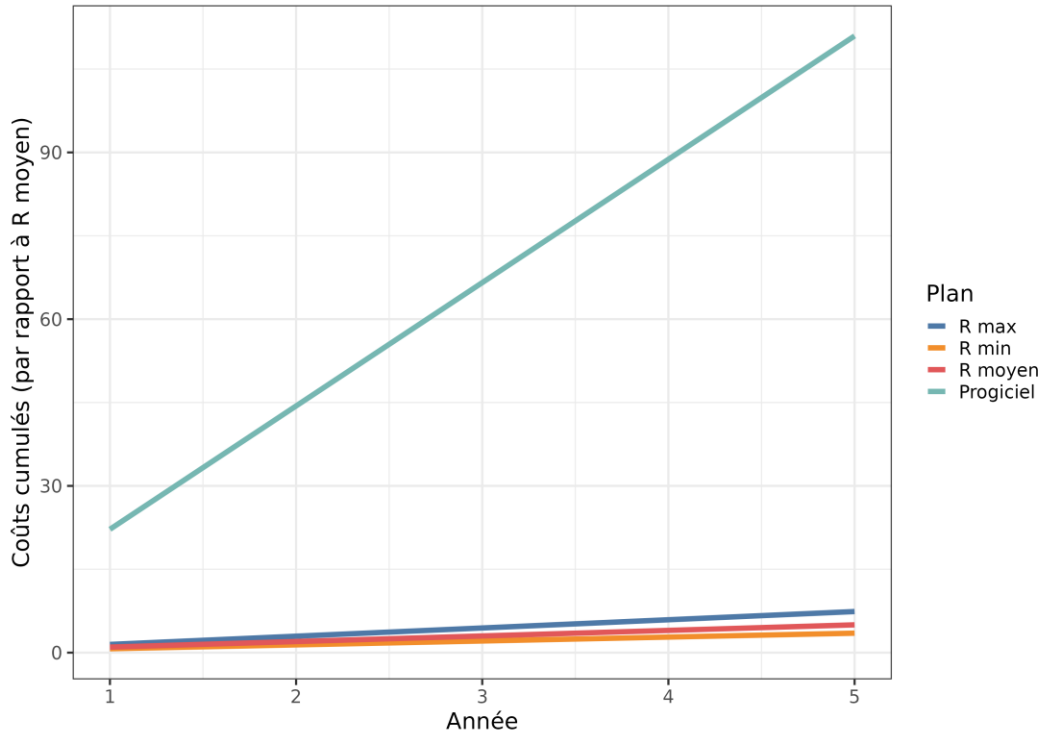


Production

- Environnement familiar
- Gestion et interface utilisateur centralisées de l'ensemble des outils
- Nombre d'actions limitées pour le paramétrage sur-mesure et l'éxécution des modèles
- Expérience utilisateur simplifiée (Excel) tout en mettant une gouvernance forte (cf slide Gouvernance)

Coûts des solutions

Action	Package R	Progiciel
Ponctuelle	• Développement de l'outil • Evolutions futures	• Développement de l'outil • Evolutions futures • Changements éditeur
Récurrente	• Serveurs (cloud)	• Maintenance • Licence annuelle / utilisateur • Serveurs (cloud)



Gouvernance

Contrôle des versions et gestion du code via GitLab

Système de tickets pour traquer les bugs et orchestrer les développements

Branche pour isoler des portions de développements

Routines automatisées de tests et de qualité de code

Documentation via Roxygen

Documentation inclue dans le code

Rendu sous forme de site web

Tests unitaires via testthat

Incite à la factorisation du code

Chaque fonction peut / doit être testée unitairement

Environnement de contrôle

Politique de contrôle des accès

Définition des rôles utilisateurs

Gestion du changement

Code source auditable

Flexibilité / Rigidité

Critère	Package R	Progiciel
Flexibilité	• Dispose de toutes les structures disponibles en R: dataframes, matrices, listes, ... • Architecture de l'outil à la main des développeurs • Factorisation du code possible	• Ne dispose que des structures mise à disposition par l'éditeur • Architecture peu flexible
Adaptabilité business	• Possibilité d'implémenter des clauses commerciales complexes • Possibilité de s'adapter finement aux contraintes règlementaires (.e.g scr rachat) • Possibilité d'intégrer très rapidement de nouveaux facteurs de risques	• Difficulté à implémenter certaines clauses contractuelles. Nécessité de recalculer a posteriori indicateurs • Implémentation potentiellement trop prudente par rapport aux contraintes règlementaires • Difficulté à intégrer de nouveaux facteurs de risques

Dette technique

Définition:
La **dette technique** est un concept utilisé en informatique pour décrire le coût à long terme de solutions rapides ou temporaires adoptées lors du développement logiciel.

Critère	Package R	Progiciel
Soutien et Support Technique	• Pas de support technique à proprement parler. En revanche une large communauté de développeurs et d'utilisateurs contribuent quotidiennement à étendre les fonctionnalités et documenter l'utilisation de R et ses packages. • Système de package et centre de compétences développé autour du langage R (propre à Partner Re). Ainsi de nombreux développeurs peuvent contribuer en cas de besoin.	• Un support technique assuré par l'éditeur. La réactivité peut varier d'un éditeur à l'autre et dans le temps • Peu de ressources documentaires facilement accessibles • Une petite communauté de développeur ayant la compétence
Évolution et Maintenance	• Les évolutions de modèles peuvent être implémentées par des membres de l'équipe interne qui maîtrisent R ou par des consultants. Dans l'équipe, au moins quatre profils ont développé des compétences sur R.	• Les évolutions de modèles peuvent être implémentées par des membres de l'équipe interne qui maîtrisent le langage du progiciel ou par des consultants. Dans l'équipe, un seul profil dispose des compétences.

Ressources humaines

Langages open source R / Python	Langages propriétaires reposant sur un progiciel
• De nombreux profils sont déjà disponibles sur le marché • Les écoles d'actuariat forment les actuaires de demain à ces langages • Ressources abondantes pour la formation des équipes • Des profils moins onéreux • Possibilité de faire intervenir des développeurs de formation dans le dispositif	• Peu de profils disponibles • Aucune école d'actuariat ne forme à l'utilisation de ces outils / langages • Peu ou pas de ressources librement accessibles pour se former en ligne • Des profils onéreux car très rares • Très difficile de faire intervenir des profils différents de ceux fournis par l'éditeur (restrictions contractuelles)

Bilan

La migration facilité par l'**intégration stratégique** de R dans l'infrastructure IT de l'entreprise et l'**expertise de l'équipe** a permis de:

- Réduire les **temps de paramétrage et de calculs** et réallouer les ressources actuarielles sur les analyses à valeur ajoutée
- Améliorer la **flexibilité** des outils de productions aux évolutions techniques et règlementaires
- Rationaliser le **processus de production** et factoriser les outils
- Améliorer l'**auditabilité, la traçabilité** et la **réplicabilité** des calculs
- Assurer la future gestion des **ressources humaines**
- Réduire l'**empreinte carbone** liée à l'utilisation des outils actuarielles
- Gagner en **indépendance** par rapport à un seul éditeur / cabinet de conseil
- Limiter les **coûts** liés au logiciel et au système informatique autour de la solution

Perspectives

Des pistes pour la suite :

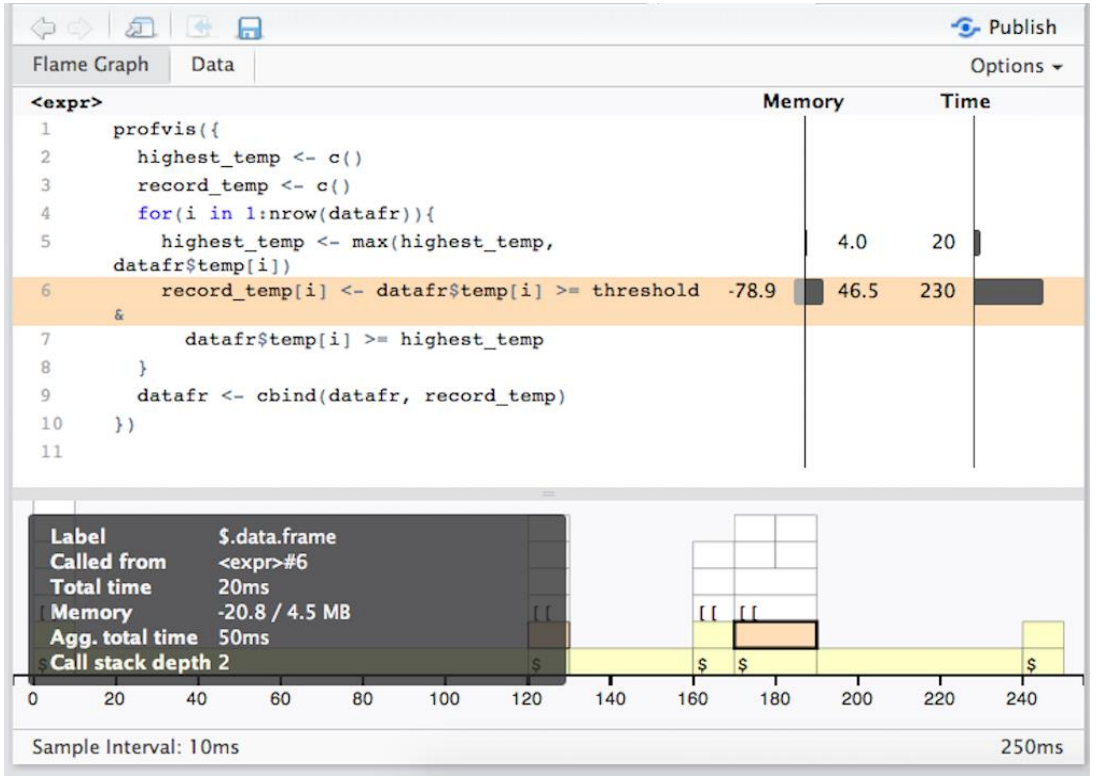
- Rationaliser le **capital économique** de l'entreprise en modélisant étant au plus près du risque et de la règlementation.
- Développer des **solutions sur mesure** en réponse aux besoins de nos clients.
- Avancer vers l'**automatisation de la chaine de calcul** (intelligence artificielle, batch).
- Repenser la **valeur ajoutée pour les clients** de la part des éditeurs de logiciels (intelligence artificielle, support client).

Annexes

Annexe: Profviz

```
record_temp_perf <- microbenchmark(find_records_1(example_data, 27),
                                   find_records_2(example_data, 27))
record_temp_perf
Unit: microseconds
                             expr      min       lq      mean   median       uq
 find_records_1(example_data, 27)  114.574  136.680  156.1812  146.132  163.676
 find_records_2(example_data, 27) 4717.407 5271.877 6113.5087 5867.701 6167.709
      max neval
  593.461   100
11334.064   100
```

microbenchmark



profviz