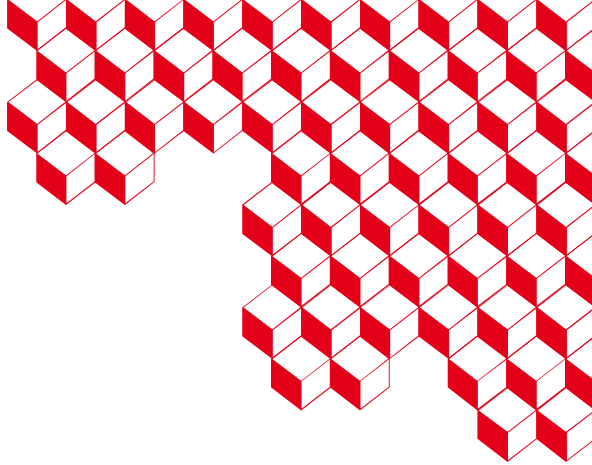


Synthèse des travaux autour de FLDLib

F. Védrine

Travaux autour de FLDLib

- Suite étude FMAS (2024)

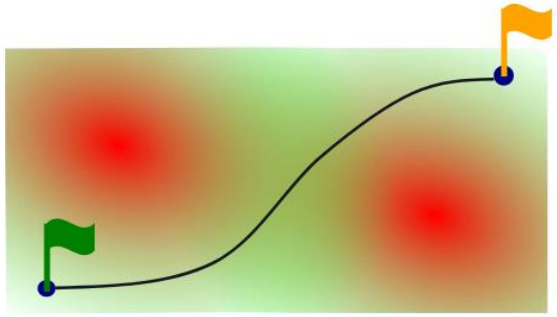
- Nouvelle version de FLDLib
  - Passage à c++-20
  - Intégration des domaines élaborés pour l'étude FMAS

  - Passage sous CMake
  - Nouvelle release 0.9.5 dans les prochaines semaines
  - Version 1.0 d'ici la fin de l'année avec les subdivisions avec exigences (spécifiques aux analyses conservatrices)
    - SUBDIV_REQUIRE_ERROR_BEGIN(x, res, -1e-5, 1e-5) … SUBDIV_END
  - Support multi-thread exclus dans l'immédiat

- Résumé discussion EDF
  - Version open source de FMAS en repartant de l'article original

- Positionnement actuel de FLDLib

Etude FMAS



- 5 locations dans le code source pour 351296 tests instables rencontrés (source CADNA) – premier test instable discontinu

```
y = 0.5, yMin = 0,
dy = 0.005 ~ 5×10^-3 - 10^-18
jm_res ~ 100 - 2×10^-16
```

```
double jm_res = ( y - yMin ) / dy ;
unsigned int jm = ( unsigned int ) jm_res;
```

mesh point

- RQ1 : Quelle est la précision du chemin calculé, en présence de plus de 400000 tests instables ?
  - Verrou : écart-type des exécutions : 2.68×10^{-4},
    mais différence entre la moyenne et l'exécution flottante = 1.27×10^{-3}
  - Cadna : écart-type des exécutions : 1.32×10^{-15},
    mais différence entre la moyenne et l'exécution flottante = 1.27×10^{-3}
  - FLDLib : impact des deux derniers tests instables rencontrés dans le calcul arrière de chemin : 4.71×10^{-13}

- Lien perdu entre propagation avant et arrière – les calculs liés au maillage sont identiques entre les deux algos (utiliser la nouvelle version de Verrou)

Identification des tests instables à fort impact

- Rajout d'une backtrace dans la sortie de Cadna à chaque rencontre d'un test instable
  - Analyse de Cadna bien plus rapide que FLDLib
  - Résultat : 8 locations dans le code source sont à l'origine de tests instables discontinus répartis sur
    - Initialisation de la grille,
    - Propagation avant de l'onde,
    - Propagation arrière de l'onde.

- Evaluation de l'impact du 1er test instable – variation de la constante dy = 0.005

$y = 0.5, yMin = 0,$

$dy = 0.005 \sim 5 \times 10^{-3} - 10^{-18}$

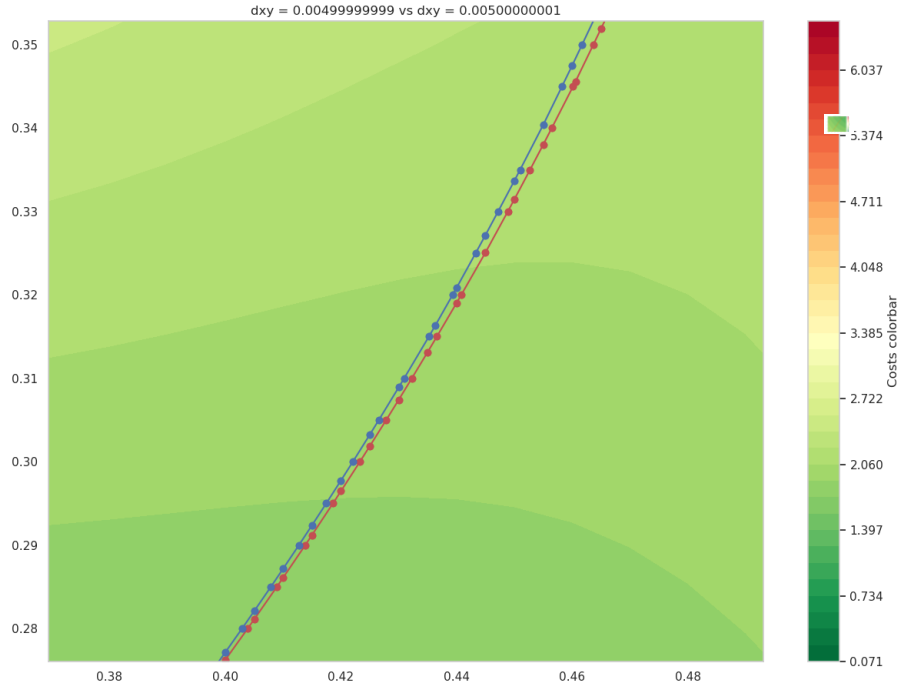
$jm\_res \sim 100 - 2 \times 10^{-16}$

```
double jm_res = ( y - yMin ) / dy ;
unsigned int jm = ( unsigned int ) jm_res;
```

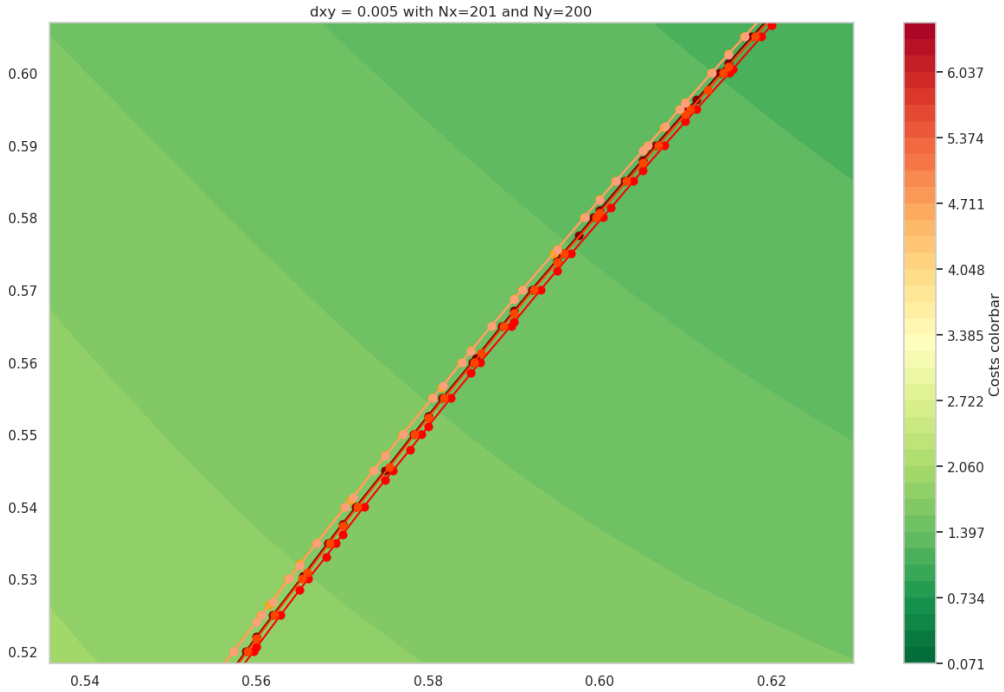
mesh point

Impact du test instable lié à l'initialisation de grille

- Des chemins différents et une erreur limitée à 1.33×10^{-3}



Chemins stables (Verrou) pour dxy en-dessous (Nx=Ny=201) ou au-dessus (Nx=Ny=200) de dxy=0.005



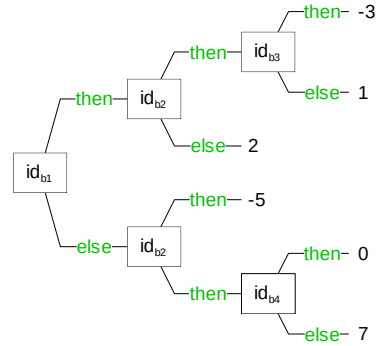
Chemins instables (Verrou) pour dxy valant 0.005

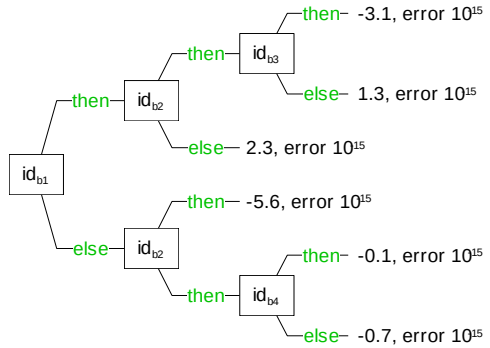
Domaines conditionnels pour identifier et qualifier précisément l'impact des tests instables

- Couverture exhaustive de tous les tests instables rencontrés pour évaluer la contribution de chacun dans l'erreur finale

- Intérêt d'utiliser Cadna
  - Analyse rapide qui détecte les tests instables
    - Il va falloir lancer l'analyse plusieurs fois pour la mise au point
  - Annotations de FLDLib déjà présentes pour synchroniser les tests
  - Mais rajout d'un domaine conditionnel dans Cadna

- Faire des analyses sur des scénarios numériques de plus en plus larges





FLDLib – passage à c++-20

- Utilisation de l'opérateur « spaceshift » <=> pour les comparaisons
  - Difficulté à déterminer comment synchroniser les tests instables if ((x <=> y) <= 0)
    - naturellement (x <=> y) a 3 valeurs : -1, 0, 1, et FLDLib a tendance à énumérer les 3 cas, dont deux (-1 et 0) vont dans la même branche avec un point de synchronisation à la fin du if
    - alors que if (x <= y) n'énumère que deux cas

- Utilisation de x-values = ok, mais a impliqué des changements dans le code existant

- Si TypeElement a un move-constructor pouvant déclencher une exception, alors lorsqu'un std::vector<TypeElement> s'agrandit, les TypeElement sont dupliqués et les anciens détruits
  - appel au copy-constructor au lieu du move-constructor

    ⇒ ne milite pas pour une programmation défensive par exception

- Mais une certaine dette technique est conservée
  - Liste chaînée au lieu de vecteurs, volonté de perdre des garanties pour gagner en vitesse d'analyse

FLDLib – Intégration des domaines optionnels et domaines conditionnels

- Domaines optionnels
  - Utilisation de copy on write à base de shared_ptr
  - Gain manifeste en temps d'analyse
  - Possibilité de delta-debugger
  - Validation en cours, régressions corrigées sur les tests intables

- Domaines conditionnels
  - A valider sur la base de tests et notamment les tests contenant des tables d'interpolation
  - Reprendre les simplifications utilisées dans les SAT-Solvers – problématique de passage à l'échelle

Positionnement « industriel » de FLDLib

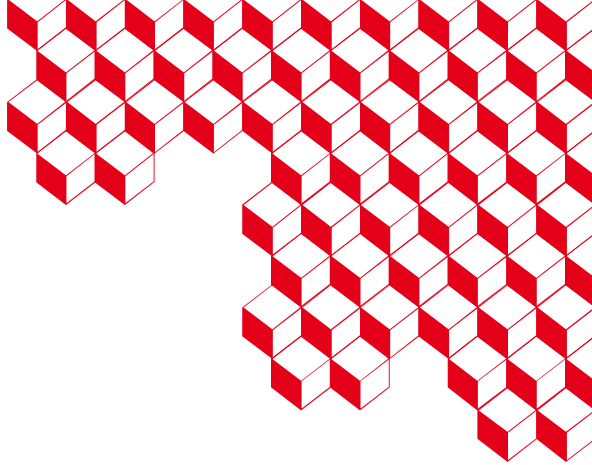
- Bibliothèque d'expérimentation capable de calculer une erreur précise sur des scénarios fins
  - Offre des « garanties », mais sur des scénarios de tests
  - A expérimenter pour se l'approprier et définir un éventuel usage associé à l'étude
- Expérimentations avec UPVD – ex analyse chromatique (mode plus data-flow)
- A appliquer après Verrou, Verificarlo et Cadna, analyse différentiée
- Cross-fertilisation Fluctuat <-> FLDLib

Positionnement « académique » de FLDLib

1. Bibliothèque d'expérimentation capable de calculer une erreur précise sur des scénarios fins
   - Possibilité de remplacer les coefficient « mpfr » des formes affines par des variables symboliques
   - Elargissement progressif des scénarios fins
     - Inférence d'une expression data-flow en fin d'analyse par rapport à ces variables
     - Inférence d'un code data et control-flow en fin d'analyse par rapport à ces variables

2. Instrumentation extensible capable de faire une preuve formelle si l'utilisateur maîtrise ce qu'il fait
   - Fournir une bonne abstraction formelle pour les container STL dans le cadre de leur utilisation
     - Quantification d'ordre supérieur
   - Prouver la validité de cette abstraction, preuves par récurrence à faire par instrumentation

3. Générer des formules, des codes de calcul d'erreur de précision, pour raisonner dessus. Inférence de formule de probabilité ?



Merci !