UNIVERSIDADE DA CORUÑA

Escola Politécnica Superior. Ferrol.

INGENIERÍA INDUSTRIAL

DESARROLLO DE SOFTWARE MODULAR, ABIERTO Y COLABORATIVO PARA SIMULACIÓN DE MÁQUINAS Y MECANISMOS

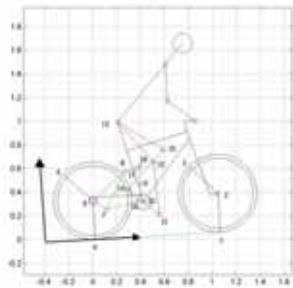
Autor: **Miguel Álvarez Loira**
Tutor: **Manuel Jesús González Castro**

Universidad de A Coruña

Ferrol, Julio 2006

Contenido

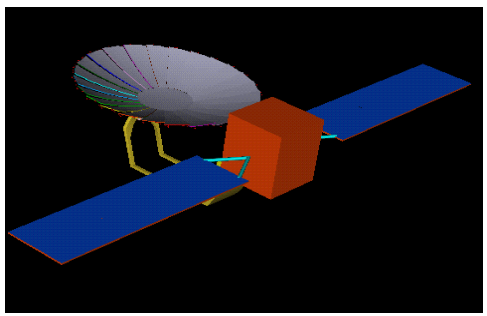
- Definición
- Laboratorio de Ingeniería Mecánica de la UDC

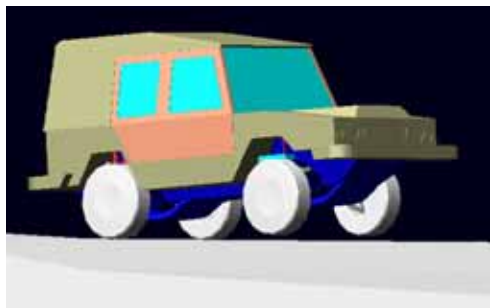


PFC: "*Diseño de un Benchmark para evaluar las prestaciones de software de simulación de sistemas multicuerpo*"

Tesis: "*Formulaciones semi-recursivas y de penalización para la dinámica en tiempo real de sistemas multicuerpo*"

Tesis: "*A Collaborative Environment for Flexible Development of MBS Software*"





Software comercial

- Alto coste
- Licencias Comerciales
- Reducida Flexibilidad
- No válido para investigación

MbsLab (*Multibody Systems Laboratory*)

Modular Abierto Colaborativo

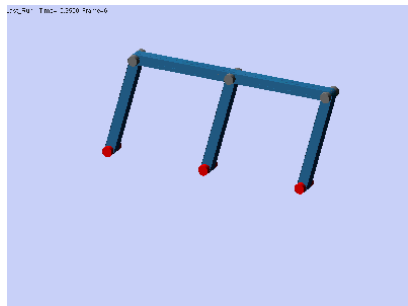
- Evaluar las distintas librerías de álgebra lineal existentes en el mercado.

- Mejorar el software de simulación dinámica MbsEngine, añadiendo más formulaciones e integradores.

- Validar las formulaciones e integradores programados.

boost C++ LIBRARIES

$$\mathbf{M}^d\ddot{\mathbf{z}} + \mathbf{\Phi}_{\mathbf{z}}^{\mathrm{T}}\boldsymbol{\lambda}^* + \mathbf{\Phi}_{\mathbf{z}}^{\mathrm{T}}\alpha\mathbf{\Phi} = \mathbf{Q}^d$$
$$\boldsymbol{\lambda}_{i+1}^* = \boldsymbol{\lambda}_i^* + \alpha\mathbf{\Phi}_{i+1}, \quad i = 0,1,2,\ldots$$
$$\dot{\mathbf{z}}_{n+1}^i = \frac{2}{h}\mathbf{z}_{n+1}^i + \hat{\dot{\mathbf{z}}}_n^i$$
$$\ddot{\mathbf{z}}_{n+1}^i = \frac{4}{h^2}\mathbf{z}_{n+1}^i + \hat{\ddot{\mathbf{z}}}_n^i$$



2. Librerías

1. Introducción
2. **Librerías**
3. Formulaciones e Integradores
4. Validación
5. Conclusiones

La mayor parte del tiempo de computación que requiere hacer una simulación dinámica se usa en resolver problemas de álgebra lineal.

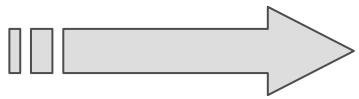
Resolver el sistema no lineal	35%
Formar la matriz tangente	33,5%
Resto	31,5%

Clasificación de las librerías utilizadas:

- **Estructuras de datos**: librería uBLAS para vectores y matrices y otras librerías con diversas utilidades.
- **Rutinas de soporte**: los BLAS o conjunto de núcleos (rutinas) computacionales.
- **Métodos directos de solución** de ecuaciones con almacenamiento denso y con almacenamiento disperso.
- **Librerías de integradores**.

Multitud de estructuras de datos para operaciones de álgebra lineal.

- Newmat
- MTL
- Blitz++
- uBLAS de Boost



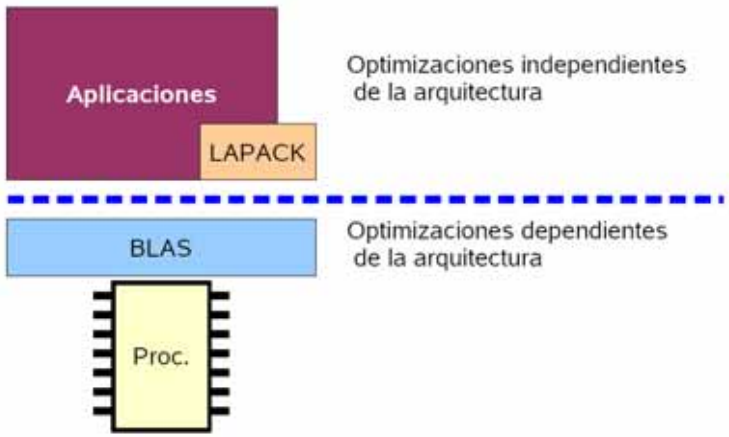
Motivos:

- Soporta matrices densas y algunos tipos de disperso.
- Es gratuita y tiene una licencia flexible no como GNU.
- Respaldo y el prestigio de Boost.

- **uBLAS** :librería numérica que proporciona plantillas de clases C++
- El diseño e implementación se hace con la notación matemática sobrecargando el operador y la generación del código mediante plantillas.

BLAS (*Basic Linear Algebra Subprograms*) conjunto de rutinas escritos en Fortran 77 para operaciones básicas que proporcionan los estándares para realizar este tipo de operaciones de álgebra lineal.

Posibilidad de usar librerías que implementan las rutinas de BLAS, y que además están optimizadas para cada arquitectura.



Algunas suministradas por fabricantes:

ACL de AMD,
Velocity Engine de Apple,
libsci de Cray,
MLIB de Hp,
MKL de Intel
SCSL de SGI, etc

Otras

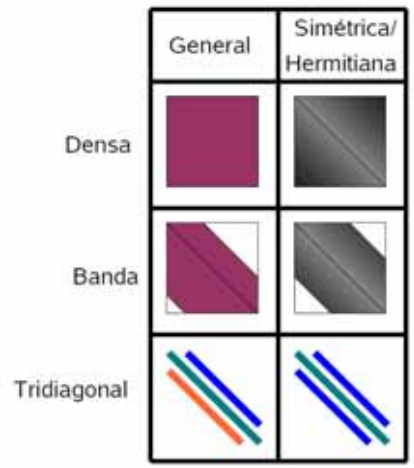
GotoBLAS,
ATLAS,
ALPHA EV,
...

Multitud de solvers para denso:

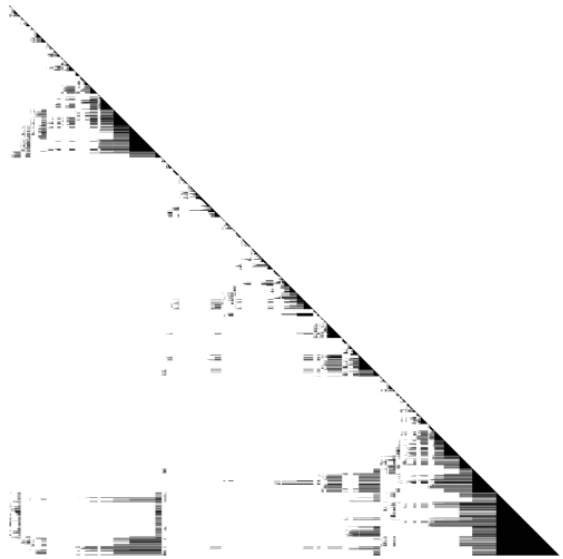
LAPACK, FLAME, LINALG, MTL, NEWMAT, GSL, y las antes mencionadas ATLAS.

ATLAS además de incorporar las rutinas de BLAS proporciona un subconjunto de especificaciones y rutinas de LAPACK para denso.

LAPACK (*Linear Algebra PACKage)* es una biblioteca que resulve sistema de ecuaciones lineales y proporciona factorizaciones asociadas a matrices LU, Cholesky, QR… Se apoya en las rutinas de BLAS.



Sparse: matrices que contienen una gran cantidad de ceros en sus términos (*nonzeros*).



- No existe un método único para representar una matriz dispersa, a diferencia del denso.

- Se pueden clasificar en librerías de matrices dispersas (SparseKIT, SparseBLAS, Boost uBLAS) y los solvers directos para disperso (CHOLMOD, TAUCS y UMFPACK).

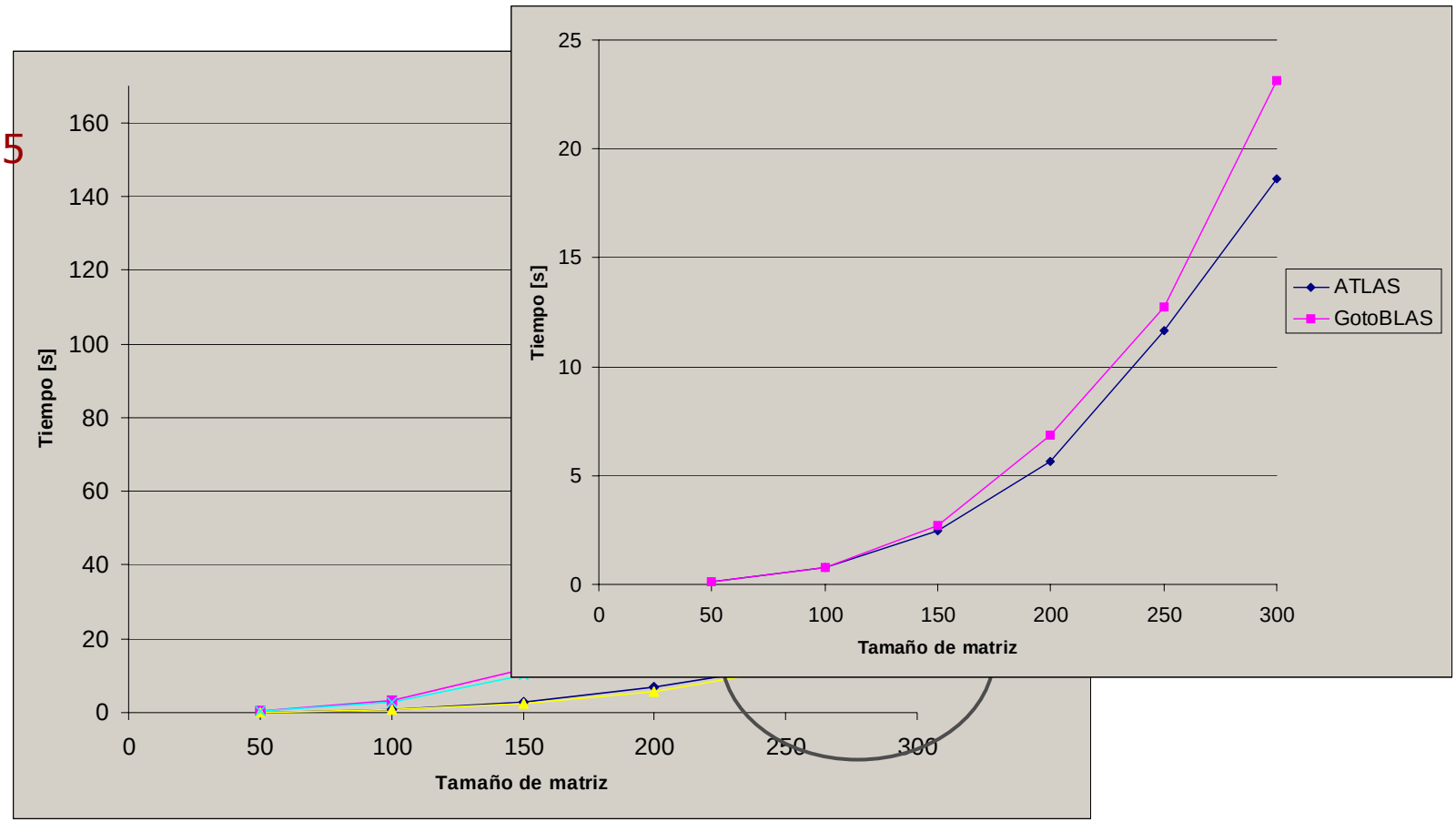
CHOLMOD es un conjunto de rutinas en ANSI C que da la posibilidad de realizar factorizaciones de Cholesky, obtener la solución de sistemas triangulares…

UMFPACK es un sistema de rutinas para resolver los sistemas lineales, Ax = b, donde A es una matriz dispersa y asimétrica

Benchmark para matrices densas en un AMD Athlon(tm) 64 3000+

ATLAS x5,5

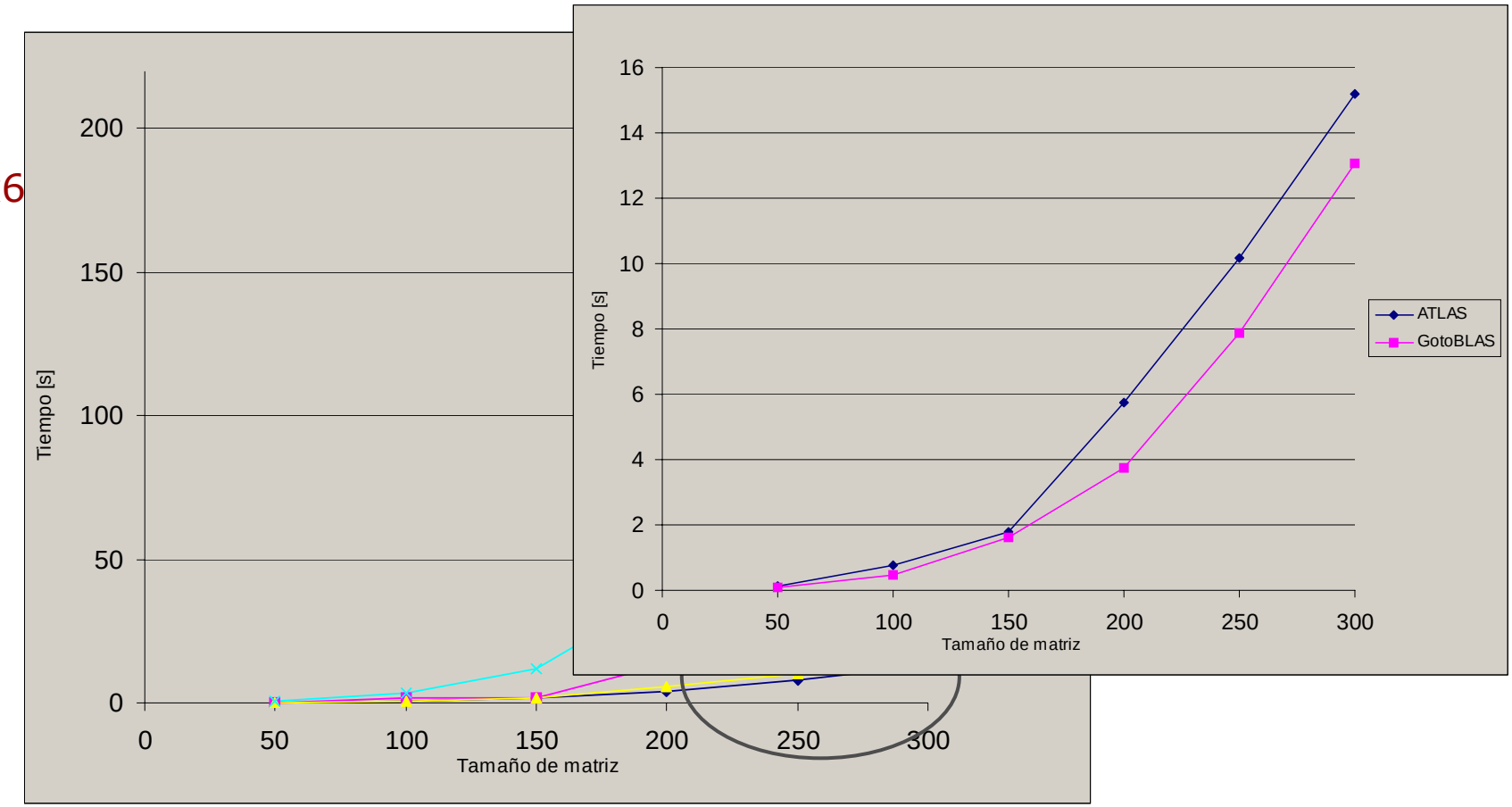
Goto x4,5



Benchmark para matrices densas en un P4

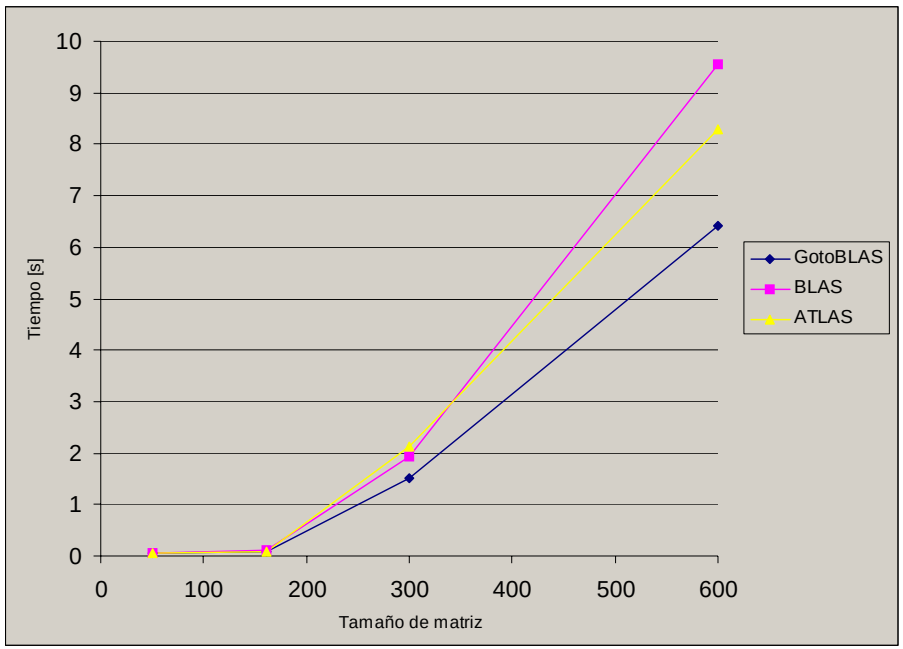
ATLAS x6

Goto x7

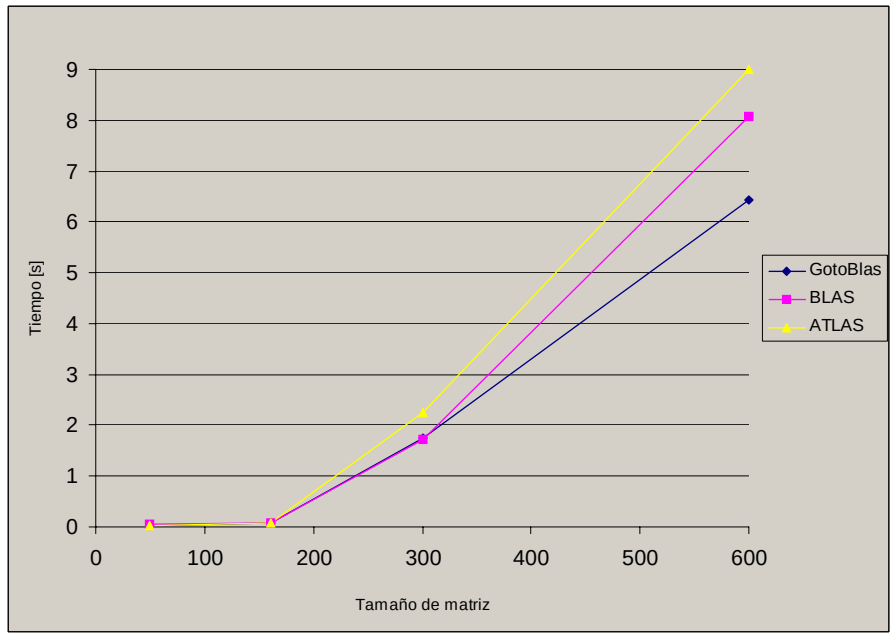


Benchmark para matrices sparse

AMD Athlon(tm) 64 3000+



P4



3. Formulaciones e Integradores

1. Introducción
2. Librerías
3. **Formulaciones e Integradores**
4. Validación
5. Conclusiones

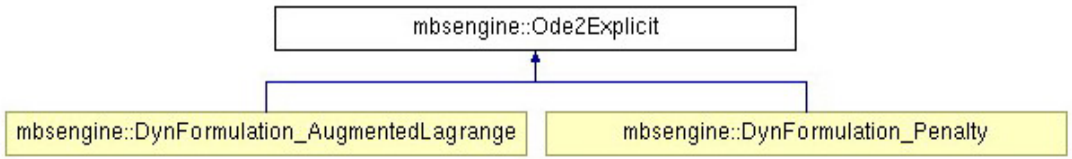
- Penalizadores

$$\left(\mathbf{M}+\boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\alpha\boldsymbol{\Phi}_{\mathbf{q}}\right)\ddot{\mathbf{q}}=\mathbf{Q}(\mathbf{q},\dot{\mathbf{q}})-\boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\alpha\left(\dot{\boldsymbol{\Phi}}_{\mathbf{q}}\dot{\mathbf{q}}+\dot{\boldsymbol{\Phi}}_{t}+2\xi\omega\dot{\boldsymbol{\Phi}}+\omega^{2}\boldsymbol{\Phi}\right)$$

- Lagrange Aumentado

$$\mathbf{M}\ddot{\mathbf{q}}+\boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\boldsymbol{\lambda}^{*}+\boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}\alpha\left(\ddot{\boldsymbol{\Phi}}+2\xi\omega\dot{\boldsymbol{\Phi}}+\omega^{2}\boldsymbol{\Phi}\right)=\mathbf{Q}(\mathbf{q},\dot{\mathbf{q}})$$

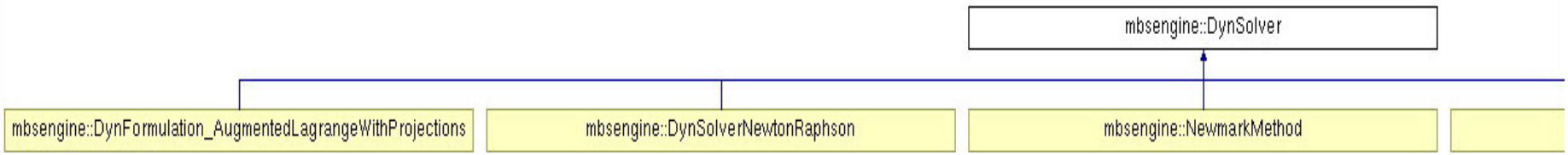
$$\boldsymbol{\lambda}_{i+1}^{*}=\boldsymbol{\lambda}_{i}^{*}+\alpha\left(\ddot{\boldsymbol{\Phi}}+2\xi\omega\dot{\boldsymbol{\Phi}}+\omega^{2}\boldsymbol{\Phi}\right)_{i+1},\quad i=0,1,2,\ldots$$



- Regla Trapezoidal
- Newmark

$$\mathbf{q}_{n+1}=\mathbf{q}_{n}+h\dot{\mathbf{q}}_{n}+\frac{h^{2}}{2}\left\{(1-2\beta)\ddot{\mathbf{q}}_{n}+2\beta\ddot{\mathbf{q}}_{n+1}\right\}$$

$$\dot{\mathbf{q}}_{n+1}=\dot{\mathbf{q}}_{n}+h\left\{(1-\gamma)\ddot{\mathbf{q}}_{n}+\gamma\ddot{\mathbf{q}}_{n+1}\right\}$$



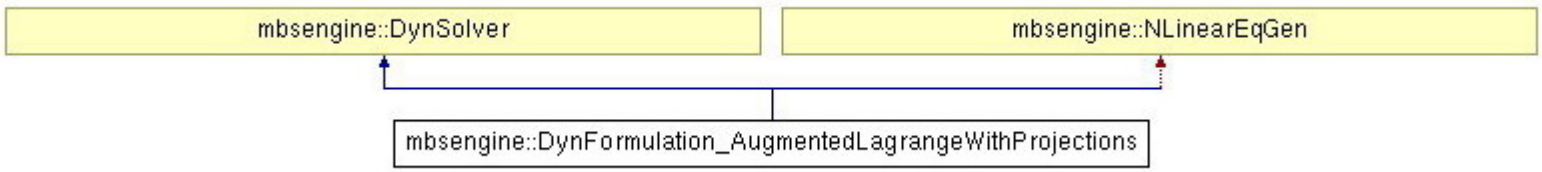
- Lagrange aumentado con proyecciones en Index-3 + regla trapezoidal

$$\left[\frac{\partial \mathbf{f}(\mathbf{q})}{\partial \mathbf{q}}\right]_i \Delta \mathbf{q}_{i+1} = -\left[\mathbf{f}(\mathbf{q})\right]_i$$

$$\left[\frac{\partial \mathbf{f}(\mathbf{q})}{\partial \mathbf{q}}\right] \cong \mathbf{M} + \frac{h}{2}\mathbf{C} + \frac{h^2}{4}\left(\mathbf{\Phi}_\mathbf{q}^{\mathrm{T}} \alpha \mathbf{\Phi}_\mathbf{q} + \mathbf{K}\right)$$

$$\mathbf{q}_{i+1} = \mathbf{q}_i + \Delta \mathbf{q}_{i+1}$$

$$\mathbf{f}(\mathbf{q}) = \frac{h^2}{4}\left(\mathbf{M}\ddot{\mathbf{q}} + \mathbf{\Phi}_\mathbf{q}^{\mathrm{T}} \boldsymbol{\lambda}^* + \mathbf{\Phi}_\mathbf{q}^{\mathrm{T}} \alpha \mathbf{\Phi} - \mathbf{Q}\right)_{n+1}$$



- Coordinar las tareas de programación

- Hosting gratuito
- Listas de correo propias
- Servidor SVN

- Software de control de versiones

Herramienta Version Control with Subversion (SVN) y el cliente Tortoise SVN

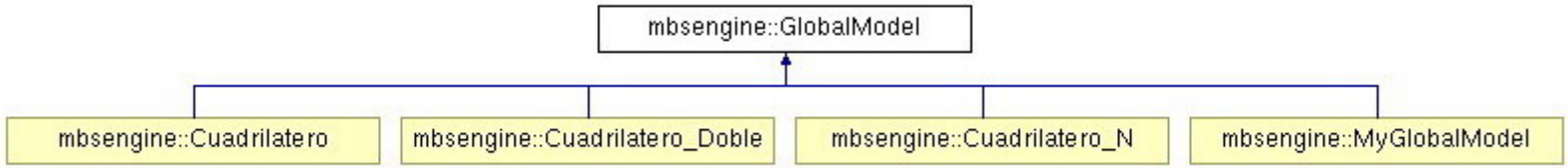
- Comparar diferentes versiones de archivos.
- Solicitar historia completa de los cambios.
- Obtener "foto" histórica del proyecto.

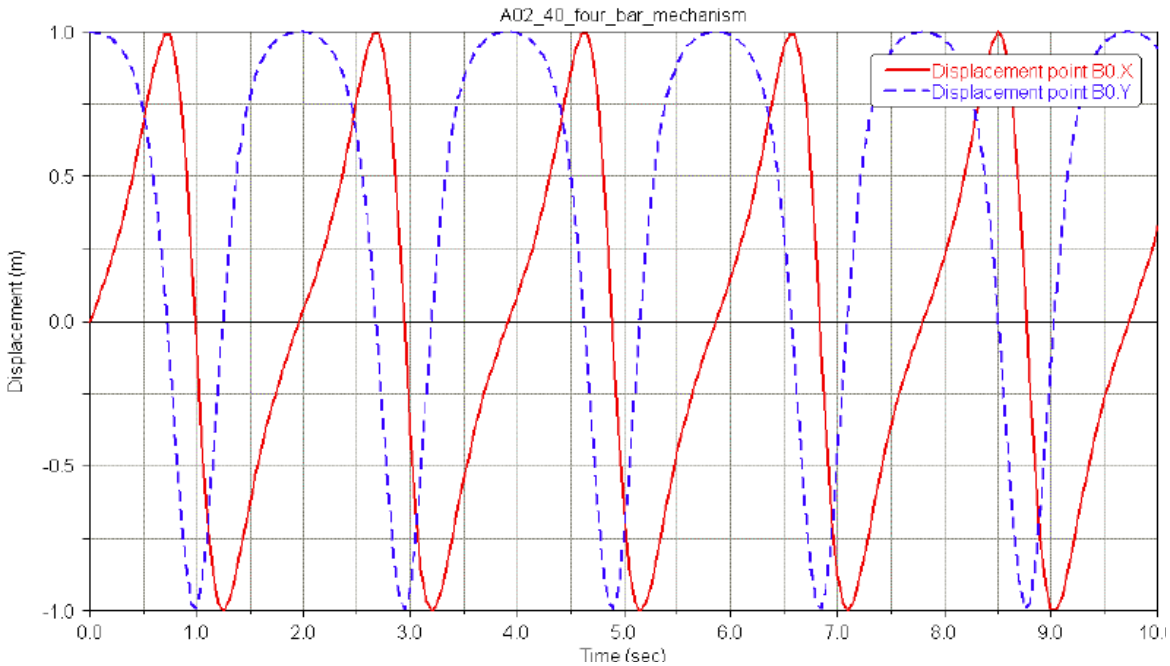
- Documentación gráfica

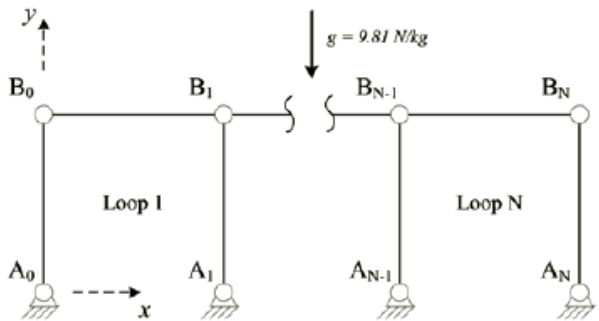
Sistema de documentación de C++, Java, IDL

4. Validación

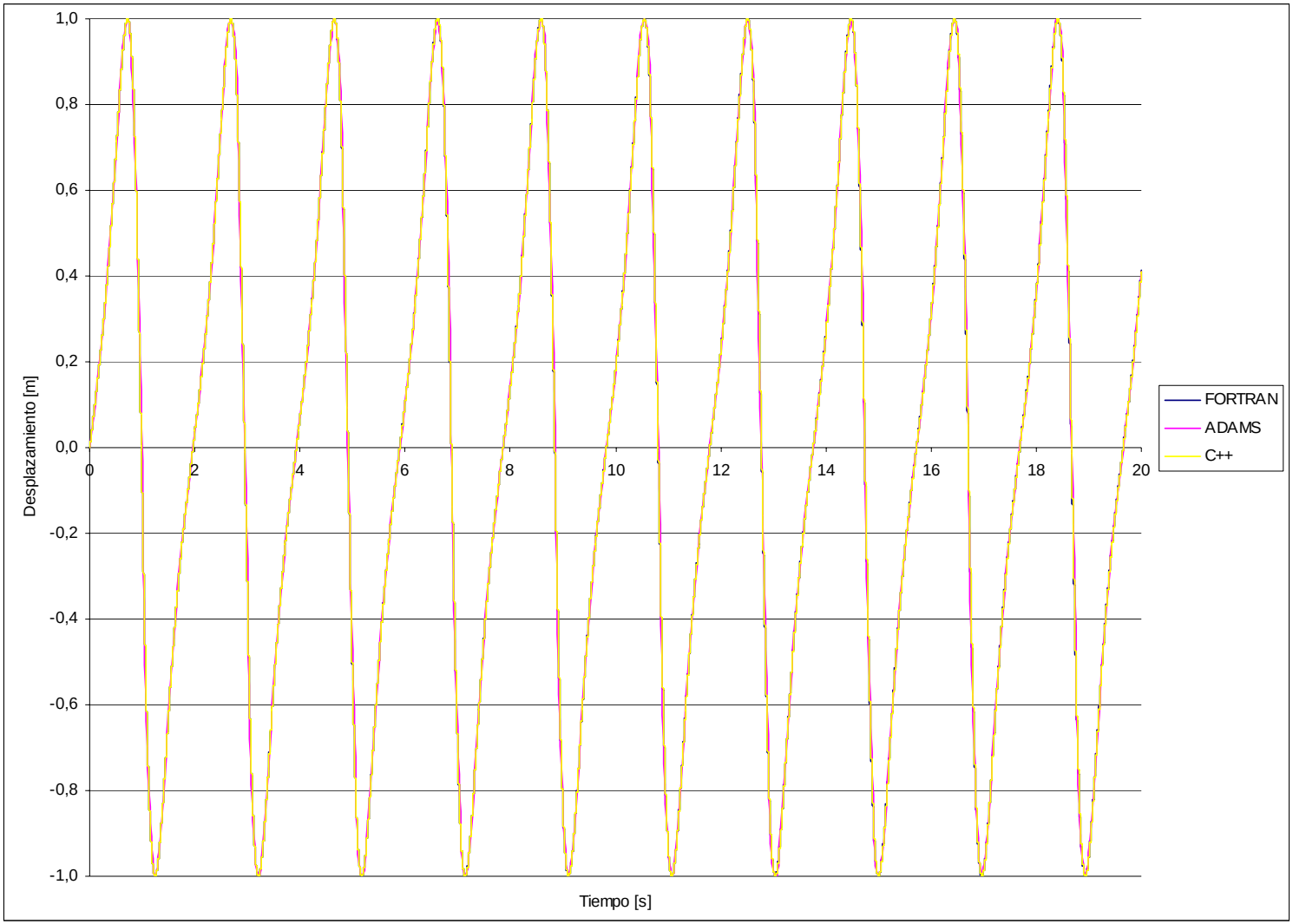
1. Introducción
2. Librerías
3. Formulaciones e integradores
4. **Validación**
5. Conclusiones







- Características
- Tiempo de simulación
- Condiciones iniciales
- Coordenada x del punto B0


1,0
0,8
0,6
0,4
0,2
0,0
-0,2
-0,4
-0,6
-0,8
-1,0
0
2
4
6
8
10
12
14
16
18
20
Desplazamiento [m]
Tiempo [s]
FORTRAN
ADAMS
C++

5. Conclusiones

1. Introducción
2. Librerías
3. Formulaciones e integradores
4. Validación
5. **Conclusiones**

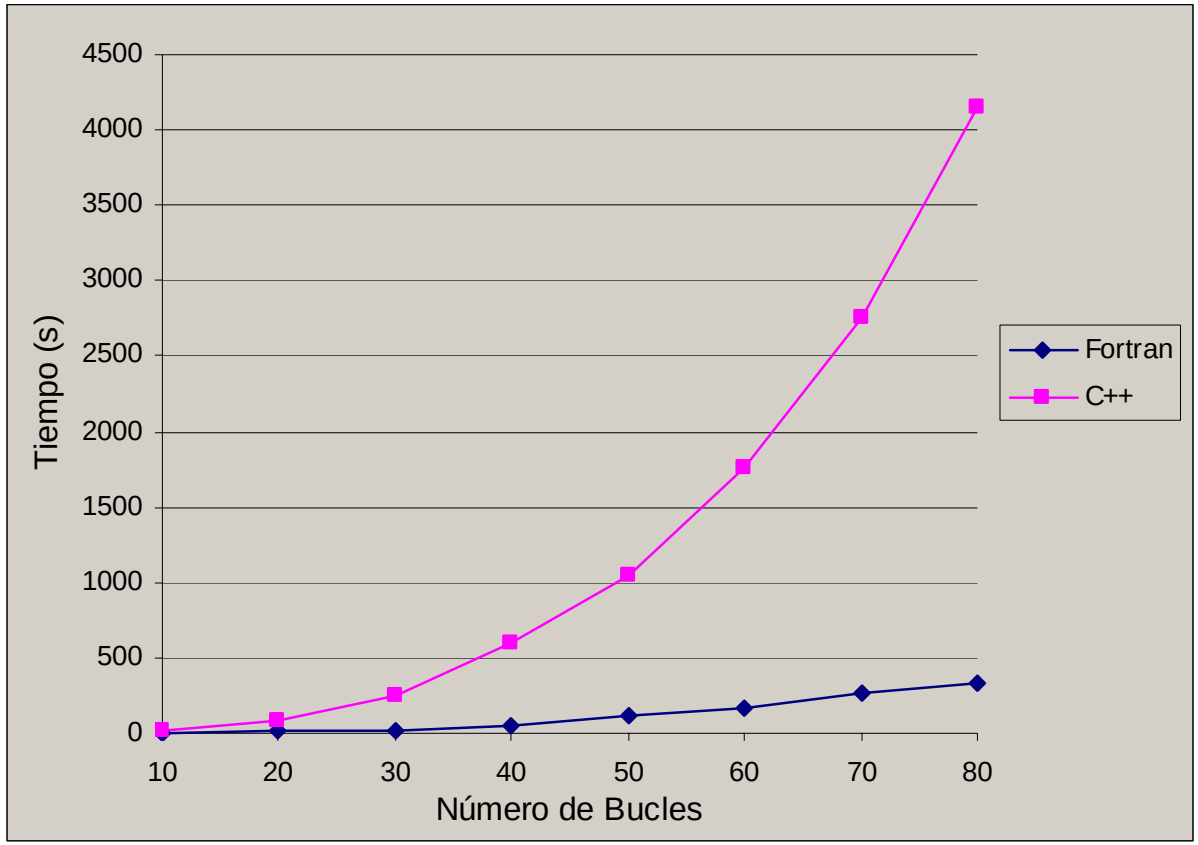
- Evaluado diferentes librerías de álgebra lineal existentes.
  - Estructuras de datos: Boost
  - BLAS e implementaciones de BLAS
  - Solvers directos para denso y sparse
- Difíciles de implementar
- Resultados son excelentes (x10)

- **Programado formulaciones e integradores en C++**
  - Dos formulaciones
    - Penalizadores
    - Lagrange Aumentado
  - Dos integradores de punto fijo
    - Newmark
    - Regla Trapezoidal
  - Combinación de formulación e integrador
    - Lagrange Aumentado con proyecciones en index-3
- **Resultados validados mediante un benchmark**

5. Conclusiones

C++ vs Fortran

- Código sin optimizar
- Usamos uBLAS
- Usamos denso



5. Conclusiones

Con librerías optimizadas, velocidad x10.

$$\frac{t_{uBLAS}}{t_{optimizado}} \approx 10$$

$$\frac{t_{uBLAS}}{t_{fortran}} \approx 10$$

$$\frac{t_{fortran}}{t_{optimizado}} \approx 1$$

- Mejorar las formulaciones e integradores programados.
  - Optimizarlas con las librerías.
  - Adaptarlas para usar librerías sparse.
- Programar más formulaciones e integradores.

UNIVERSIDADE DA CORUÑA

Escola Politécnica Superior. Ferrol.

INGENIERÍA INDUSTRIAL

DESARROLLO DE SOFTWARE MODULAR, ABIERTO Y COLABORATIVO PARA SIMULACIÓN DE MÁQUINAS Y MECANISMOS

Autor: **Miguel Álvarez Loira**
Tutor: **Manuel Jesús González Castro**

Universidad de A Coruña

Ferrol, Julio 2006