

Alternatives lliures pel càlcul científic i tècnic

Jordi Saludes

23 de maig de 2003

Dpt. matemàtica aplicada 2,
Universitat Politècnica de Catalunya

Moltes vegades, a l'encarar un projecte amb bona part de càlcul científic és convenient desenvolupar un prototip en un llenguatge no compilat per tal de fer-se una idea inicial de les dificultats del projecte, o bé per tal d'experimentar amb comoditat diverses alternatives.

Generalment ens decantem per un llenguatge de guió (*script language*) de caràcter comercial. A la primera part d'aquest paper resumirem les opcions comercials generalistes més usades pel càlcul tècnic. A la segona part mostrarem una experiència d'acostament a la implementació d'un projecte d'aquest estil utilitzant programari lliure.

1 Càlcul científic

Les necessitats de càlcul científic solen caure en les següents categories:

Càlcul numèric. On es manipulen seguits homogenis de números: vectors, matrius o tensors d'ordre superior. Hi ha diversos llenguatges especialitzats en aquest àmbit, però indubtablement el més popular és *Matlab* [3].

Càlcul simbòlic. En aquest cas es treballa amb expressions matemàtiques que poden incloure paràmetres o incògnites representats amb símbols. Tot i aquest enfoc, aquests sistemes poden descendir al reialme numèric si es dona el cas. Els sistemes més coneguts de càlcul simbòlic són *Maple* [6] i *Mathematica* [5].

1.1 *Matlab*

Pel càlcul numèric el sistema més estès és *Matlab*. Es tracta d'un sistema pel càlcul numèric que va créixer a partir de les biblioteques de rutines d'àlgebra lineal *LAPACK* i *BLAS* [7, 8].

Matlab té un llenguatge interpretat propi que dona accés a les rutines numèriques de les biblioteques esmentades. Una de les raons que, al meu entendre, ha fet extremadament popular aquest producte radica en la naturalitat

amb que aquest llenguatge permet d'expressar operacions en vectors i matrius que en altres llenguatges exigeixen un o més llaços `for`.

Amb el temps el sistema ha crescut incorporant moltes *toolboxes* adreçades al càlcul tècnic, de manera que ara és possiblement el producte amb l'oferta de càlcul més àmplia. A part d'això, val a dir que el suport de gràfics és molt complet i que permet fer també càlcul simbòlic mitjançant una pasarel·la a *Maple*. A més, l'entorn té un sistema interface que permet d'accedir a rutines en llenguatge *C* definides per l'usuari.

A la banda dels punts febles cal destacar que les estructures i classes de dades són molt elementals i donen poc joc; per exemple, els atributs dels objectes són privats i no es poden accedir sinó és a través d'un mètode.

Matlab és recomanat com a llenguatge per introduir els estudiants en el càlcul científic, però no poden usar-lo legalment a casa: Un servidor de llicències fa que només pugui usar-se des del campus i encara només en un número limitat de llicències simultànies¹.

1.2 Càlcul simbòlic

Pel que fa al càlcul simbòlic hi ha dos competidors lluitant pel favor del públic que són *Maple* i *Mathematica*. Les capacitats dels dos programes són molt semblants i representen l'anàleg del *Matlab* en l'àmbit simbòlic: Els dos tenen llenguatge propi que té característiques imperatives i funcionals, tenen una gran biblioteca d'algoritmes pel càlcul formal i també fan càlcul numèric (però pateixen quan les dimensions de les dades són grans).

Maple s'utilitza en docència a la UPC perquè la institució té una llicència que permet d'usar-lo al campus però que no permet que els estudiants el copiïn per usar-lo a casa.

1.2.1 Problemes directes i problemes inversos

Si bé al càlcul numèric les operacions no solen fallar (o bé ho fan per raons molt concretes) al càlcul simbòlic es dóna molt sovint el cas que el sistema no sap com acabar una comanda donada. Aquí cal distingir entre operacions:

Directes: com ara derivar una funció, avaluar una funció a un punt, substituir una expressió per una altra. Són operacions per les quals hi ha un algoritme estàndard per a fer-les.

Inverses: com ara trobar la primitiva d'una funció, resoldre un sistema d'equacions. En general són les operacions inverses de les anteriors, per les quals no hi ha un algoritme determinat sinó només una col·lecció d'estratègies que podrien funcionar. En aquest apartat cal posar-hi també el problema de determinar si dues expressions són matemàticament iguals; per exemple les dues expressions

$$(x - y)(x + y), \quad x^2 - y^2.$$

¹Els meus alumnes s'han quedat més d'una vegada sense pràctiques perquè no hi havia llicències lliures a l'hora de classe.

són iguals però sintàcticament diferents²

2 L'experiència

Teníem entre mans un projecte de seguiment automàtic de línies de carril en carreteres a partir de les imatges d'una càmera muntada al vehicle. És un projecte que requereix càlcul en:

- Visió per computador a baix nivell, eficient en temps; implementat com una rutina en llenguatge C.
- Manipulació formal del model matemàtic d'imatge de carretera: Cal manipular coordenades homogènies, projectivitats i derivació simbòlica.
- Àlgebra lineal numèrica pel que fa al càlcul de les matrius del filtre de Kalman
- Gràfics per a representar imatges i corbes simultàniament.

A banda d'això s'han de definir estructures de dades per als diferents models de carretera, mostres de vídeo i resultats.

2.1 Un nucli petit amb moltes extensions

Tot i que *Matlab* és més evident, atès que compleix tots els requeriments (compareu amb l'apartat 1.1) no l'hem adoptat. Com tampoc hem triat l'alternativa de codi lliure *Scilab* [9], perquè ens calia suport de classes i objectes que encara és més escàs que el qui hi ha a *Matlab*. En lloc d'això hem escollit *Python* [2]: un llenguatge de guió, orientat a objecte, amb una sintaxi senzilla que conté bones idees d'altres llenguatges i que té llistes i diccionaris com a elements nadius; però que no està dedicat especialment al càlcul científic.

Ara bé, *Python* disposa d'una extensa biblioteca de recursos lliures fets per terceres parts que es poden importar com a extensions i —tal com també passa amb *Matlab* i *Scilab*— pot connectar amb rutines C fetes per l'usuari.

2.2 Tractament d'imatges i gràfics

La manipulació de les imatges amb tècniques de visió de baix nivell es pot fer lligant la rutina C que fa l'anàlisi de la imatge usant un sistema d'embolcalls ("wrappers") que té *Python* i que permet invocar funcions C des de *Python* i emmagatzemar els apuntadors a objectes externs (imatges en aquest cas)

La presentació gràfica d'imatges i corbes es pot abordar amb *Tkinter* que és la interface *Python* a *Tk*: Una biblioteca per construir pannels i finestres prou potent com per desplegar imatges i traçar poligonals; a més de poder col·locar i eliminar aquests objectes programàticament.

²Maple 8 no reconeix aquestes dues expressions com a iguals.

2.3 Càlcul matricial i simbòlic

Les necessitats d'àlgebra lineal (filtres de Kalman) les podem cobrir afegint a *Python* el paquet *Numeric* [4] que fa l'equivalent del *Matlab* pel que fa a la manipulació de matrius i a més usant la mateixa sintaxi.

Només cal tractar l'accés a un sistema d'àlgebra simbòlica. Hem considerat dues opcions: *ginac* i *yacas* [10, 11]. La primera és una biblioteca C++ per problemes directes (vegeu l'apartat 1.2.1) que treballa amb aritmètica entera o racional i números en coma flotant de precisió indefinida, permet la manipulació simbòlica de matrius i polinomis, la diferenciació de funcions, substitució, etc. Pel que fa a *yacas* és un entorn lliure interactiu a l'estil de *Mathematica* o *Maple* i que a part dels problemes directes, resol equacions (suposadament) i fa integració simbòlica.

Tant l'un com l'altre són suficients per a la mena de càlcul simbòlic que el nostre projecte requereix. Cal pensar quin dels dos mètodes és més fàcil de connectar amb *Python*. *Ginac* es pot connectar usant les funcions embolcall de *Python* de manera similar a com ho hem fet a l'apartat 2.2, mentre que *yacas* pot ser accedit des de *Python* de dues maneres:

- A través d'unes rutines C que venen amb la distribució de *yacas* i fent la connexió com als casos que acabem d'esmentar.
- Engegant *yacas* com a servidor que escolta en un port determinat. Llavors podem usar el mòdul `socket` de *Python* per tal de passar cadenes de caràcters corresponents a expressions vàlides *yacas* per tal que siguin avaluades i retornades a l'entorn *Python*.

2.3.1 Un generador d'embolcalls

Si la connexió entre *Python* i la part programada en C implica diverses rutines és convenient utilitzar una eina més sofisticada per a fer l'enllaç: *Swig* [1] és un compilador d'interfaces que genera automàticament les rutines embolcall que permeten ajustar els tipus de dades entre un entorn de llenguatge (*Python* en aquest cas) i el món C.

Per fer la connexió amb la biblioteca *ginac* emprant *Swig* definirem dues classes C++ intermèdies `GISymbol` i `GIExpresion` que amaguen apuntadors a les classes *Ginac* `Symbol` i `Expresion`.

A la classe `GIExpresion` definim els operadors aritmètics usuals (suma, resta multiplicació i divisió) de manera que criden les funcions C++ corresponents de *ginac*. Des de *Python* les instàncies de `GISymbol` i `GIExpresion` apareixen com a objectes de classes *Python* (generades automàticament per *Swig*), i quan sumem o multipliquem instàncies d'aquestes classes, s'invoquen les operacions corresponents a *ginac*.

3 Conclusions

En molts casos, utilitzant programari lliure, es poden duplicar les funcions de programari comercial de càlcul científic i tècnic al cost d'haver de buscar a la

xarxa els mòduls adients i d'haver de fer la connexió.

Al model comercial d'"una aplicació ho fa tot" hi podem oposar el model de diverses peces especialitzades connectades amb un llenguatge de guió i amb interfaces generades automàticament. El preu que paguem fent nosaltres la connexió, el guanyem en flexibilitat.

Referències

- [1] Swig <http://www.swig.org>
- [2] Python <http://www.python.org>
- [3] Matlab <http://www.mathworks.com>
- [4] Numeric <http://www.lnl.gov/pynum>
- [5] Mathematica <http://www.wolfram.com>
- [6] Maple <http://www.maplesoft.com>
- [7] Biblioteca LAPACK <http://www.netlib.org/lapack>
- [8] Biblioteca BLAS <http://www.netlib.org/blas>
- [9] Scilab <http://www-rocq.inria.fr/scilab>
- [10] GiNAC <http://www.ginac.de>
- [11] Yacas: Yet Another Computer Algebra System <http://yacas.sourceforge.net>