# Optimización numérica

Proyecto 2. PCS

Santiago Novoa Pérez. 20 de octubre del 2015.

**Introducción**

El método de programación cuadrática sucesiva (PCS) es uno de los métodos más efectivos para resolver problemas de optimización no lineales. Para que el método funcione se necesita que tanto la función objetivo como las restricciones sean dos veces continuamente diferenciables.

Los métodos PCS resuelven una secuencia de subproblemas de optimización en la que, en cada subproblema, se intenta resolver un modelo cuadrático de la función objetivo sujeta a una linealización de las restricciones (como se vió ejemplificado anteriormente en un ejercicio del examen); puede ser utilizado tanto con un marco de regiones de confianza como uno de búsqueda lineal.

En general los métodos PCS se pueden ver como una generalización del método de Newton para optimización sin restricciones dado a la manera en que encuentran una dirección de descenso. Para este proyecto en particular, como todas las restricciones son de igualdad, el método es equivalente a aplicar el método de Newton a las condiciones KKT de primer orden del problema a optimizar.

PCS es apropiado para problemas grandes y pequeños con funciones no lineales; en estos casos la solución se alcanza en un número de iteraciones sustancialmente menor que $n$.

El problema que buscamos resolver es:

$$\begin{aligned} \text{minimizar} \quad & f(x) \\ \text{s.a.} \quad & c(x) = 0 \end{aligned}$$

$$\begin{aligned} f : \mathbb{R}^n \quad &\longrightarrow \mathbb{R} \\ c : \mathbb{R}^n \quad &\longrightarrow \mathbb{R}^m \\ \Omega = \{x \in \mathbb{R}^n \mid \quad & c(x) = 0\} \end{aligned}$$

* f, c dos veces continuamente diferenciables

**Construcción del método**

- Condiciones de Optimalidad

Sean

$$\begin{aligned} Ł(x,\lambda) &= f(x) - \lambda^T c(x) \\ A(x) &= \left[\nabla c_i^T(x)\right]_{i=1\ldots m} \end{aligned}$$

Si $x^*$ es minimizador local de $f$, entonces $\exists\ \lambda^*$ tal que:

$$\begin{aligned} \nabla_x Ł(x^*,\lambda^*) = \nabla f(x^*) - & \sum_{i=1}^m \lambda_i \nabla c_i(x^*) \\ = & \quad 0; \\ \nabla_\lambda Ł(x^*,\lambda^*) = & \quad c(x^*) \\ = & \quad 0; \end{aligned}$$

Estas ecuaciones resultan de intentar ver en dónde el gradiente de la función lagrangiana se anula para poder encontrar el mínimo de esa función (condiciones necesarias de primer orden).

Si además los gradientes de las restricciones son L.I. (complementariedad estricta), entonces:

$$w^T \nabla_{xx} Ł(x^*,\lambda^*) w \ \geq\ 0 \quad \forall w \in C(x^*,\lambda^*)$$

con $C(x^*,\lambda^*)$ el cono crítico en $x^*$ , $\lambda^*$, que son todas las direcciones que cumplen con que mantienen las restricciones (en una aproximación lineal) cercanas a factibilidad, es decir:

$$C(x^*,\lambda^*) = \{w \in \mathbb{R}^n \mid \nabla c_i(x^*)^T w = 0 \ \ \forall i \in \{1,...,m\}\}$$

Lo que esto asegura es que cuando el gradiente sí se anule,o la dirección que se tome sea de descenso o todas las direcciones sean de ascenso ( usando la expansión de Taylor y el Lema de Farkas: estar en el cono ( con su respectivo $\lambda$) o que exista descenso). También se les conoce como condiciones necesarias de segundo orden.
Si queremos asegurar que el problema (convexo) va a encontrar su mínimo en $x^*$ (condiciones suficientes de segundo orden), tenemos que ver que la Hessiana de la matriz Lagrangiana sea positiva definida en la proyección de las direcciones del cono crítico:

$$0 < w^T \nabla_{xx} Ł(x^*,\lambda^*) w \quad \forall\ w \in\ \{C(x^*,\lambda^*) \setminus w = 0\}$$

- Método cuadrático

La matriz de KKT del problema anterior queda expresada como sigue:

$$K = \begin{bmatrix} W_k & A^T(x_k) \\ A(x_k) & 0 \end{bmatrix}$$

donde $W_k$ es la matriz Hessiana de la función Lagrangiana del problema de optimización.

La K es un resultado de escribir las condiciones de optimalidad descritas anteriormente en un sistema de ecuaciones:

$$\begin{bmatrix} W_k & A^T(x_k) \\ A(x_k) & 0 \end{bmatrix} * \begin{bmatrix} h_x \\ -\lambda_{k+1} \end{bmatrix} = - \begin{bmatrix} \nabla f(x_k) \\ c(x_k) \end{bmatrix}$$

Es fácil verificar que resolver el sistema anterior es equivalente a resolver otro subproblema cuadrático:

$$\begin{aligned} min \quad & \frac{1}{2} p^T Q_k p + g_k^T p \\ s.a. \quad & A_k^T p + c_k = 0 \end{aligned}$$

Aquí parece que se ha perdido el significado inicial del problema dado que se empezó hablando de $\nabla_{xx}$Ł$(x_k)$ y de si su proyección en el espacio nulo de $A(x_k)$ (todas las direcciones en $C(x_k, \lambda_k)$ con k lo suficientemente grande para que $x_k$ y $\lambda_k$ se encuentren en el cono factible (una vecindad factible de la solución)) era positiva definida, sin embargo, gracias al resultado de un artículo de Nicolas I.M. Gould [1], este nuevo problema de optimización tiene propiedades deseables. En primer lugar, si la proyección de la Hessiana de la Lagrangiana sobre el espacio nulo de los gradientes de las restricciones es SPD, la solución de los dos modelos existe y es única. También se prueba que el mínimo del problema de optimización se puede calcular usando una base del espacio nulo. Sin embargo, el resultado más importante (por su practicidad) de la lectura vincula a la inercia de la Hessiana proyectada sobre el nulo con la inercia de K.

$$\begin{gathered} Z_k^T \nabla Ł(x_k, \lambda_k) Z_k = Z_k^T W_k Z_k > 0 \\ \implies \quad \exists!\ p \ \ t.q.\ \text{p minimiza el problema cuadrático} \\ \therefore p \text{ minimiza al problema original} \\ inercia(K) = inercia(Z^T W_k Z^T) + (m, 0, m) \\ \therefore inercia(K) = (m, 0, n) \iff Z^T W_k Z^T \text{ es SPD} \end{gathered}$$

La inercia de una matriz es una ternia que dice el número de valores propios menores a cero, cero y mayores a cero respectivamente :

$$inercia(A) = (\lambda_-, \lambda_0, \lambda_+)$$

Todos estos resultados nos aseguran que, si la inercia de la matriz K es la correcta $(m, 0, n)$, resolver el problema de minimización con restricciones de igualdad formulado al principio de este trabajo es equivalente a ir resolviendo una serie de subproblemas cuadráticos convexos representados por las ecuaciones de KKT. Paso a paso será necesario resolver, para cada matriz K, el correspondiente vector h que resulte en el lado derecho de la ecuación.

[1] **Nicolas I.M. Gould**; *On practical conditions for the existence and uniqueness of solutions to the general equality quadratic programming problem*

La solución (por ser un problema convexo), si las condiciones anteriores se cumplen, existirá paso a paso y será única.

$$K * h_k = LD = - \begin{bmatrix} \nabla f(x_k) \\ c(x_k) \end{bmatrix}$$

Este proceso parece ser igual de complicado que resolver el problema de minimización del inicio, sin embargo, gracias a métodos, como el programado en matlab llamado ldl, que factorizan matrices en matrices de bloques diagonales, es posible encontrar la inercia de una matriz y resolver sistemas de ecuaciones lineales sin que el costo computacional sea tan alto.

### Convergencia global

Ya que sabemos que existe un proceso que podemos seguir para resolver nuestro problema de optimización con restricciones de igualdad, falta ahora construir algún algoritmo que nos garantice que el proceso va a converger en un número finito de iteraciones sin importar del punto inicial (convergencia global).

Se sabe que en problemas de optimización sin restricciones es posible asegurar que un método converge añadiéndole condiciones de Wolfe o regiones de confianza. El problema ahora es que para que un método sea práctico tenemos que asegurar que al ir progresando hacia un mínimo no perderemos factibilidad.

Para lograr nuestro objetivo se empezará definiendo una nueva función llamada **función de mérito**:

$$\phi(x, \mu) = f(x) + \mu \|c(x)\|$$

La función de mérito consta de dos partes, la función objetivo inicial y alguna norma de las restricciones multiplicadas por un parámetro de penalización. Lo que se intenta con la función de mérito es balancear nuestros dos objetivos:

1. Minimizar la función objetivo
2. Conservar factibilidad

Aunque existen muchas normas diferentes para la penalización del problema que nos resultarían útiles gracias a las propiedades que conllevan, en nuestro problema utilizaremos la norma uno:

$$\phi(x, \mu) = f(x) + \mu \|c(x)\|_1$$

En particular, esto nos servirá para asegurar que la función de mérito es una función exacta[2]. Que la función de mérito sea exacta asegura que al ir minimizando $\phi(x, \mu)$ también iremos encontrando la solución de nuestro problema de minimización inicial ($f(x)$), sin embargo, haber elegido la norma uno nos enfrenta a otro problema con el que no habíamos tratado anteriormente. Siempre habíamos sido capaces de derivar la función objetivo para buscar su

[2] **Nocedal, Wright**; *Numerical Optimization*

mínimo, ahora eso no será posible dado que la penalización del problema no es diferenciable.

En problemas de minimización sin restricciones, la diferenciabilidad de la función objetivo nos ofrecía una manera fácil de garantizar que una dirección se podía tomar buscando descenso desde cualquier punto (si el punto era el minimizador, la dirección de descenso era no moverse). Como no hay diferenciabilidad en la función de mérito, buscaremos otra manera de aceptar una dirección de descenso. Si se piensa de otra manera la dirección de descenso en los problemas de minimización sin restricciones, se puede ver que en realidad no era necesaria la diferenciabilidad de la función, más bien lo que se hacía era usar la derivada direccional en cada punto para ver cuál era todo el conjunto de direcciones que garantizaban descenso (usando el grandiente de la función). La ventaja de esta forma de ver al descenso es que a nuestra función $\phi$ sí se le puede obtener algún tipo de derivada de direccional o una aproximación.

El enfoque que se tomará para hacer el método será el mismo que el de una búsqueda lineal, en cada paso primero se verificará que exista descenso (El parámetro de penalización $\mu$ asegurará esto al estar acotado por la derivada direccional*) y después se involucrará a la primera condición de Wolfe para acortar el paso asegurando que el descenso sea suficiente (se aade un parámetro $\alpha$ que asegura la suficiencia.):

$$\begin{aligned} D(\phi(x_k, \mu_k); p_k) = & \quad \nabla f_k^T p_k - \mu \|c_k\|_1 \\ D(\phi(x_k, \mu_k); p_k) \leq & \quad -p_k^T W_k p_k - (\mu - \|\lambda_{k+1}\|_\infty)\|c_k\|_1 \end{aligned}$$

**Experimento**

1. Constantes $c_1 = 10^{-4}$ , $\rho = \frac{1}{10}$ , $maxiter = 100$ , $TOL = 10^{-7}$
2. Condiciones de paro

$$\begin{aligned} ||c(x_k)||_\infty \leq & \quad TOL(1 + ||c(x_0)||_\infty) \\ ||\nabla \text{Ł}(x_k, \lambda_k)||_\infty \leq & \quad TOL(1 + ||\nabla \text{Ł}(x_0, \lambda_0)||_\infty) \end{aligned}$$

3. Reporte del experimento

Cuadro 1: **PCS global** (*con función de mérito*)

| **problema** | **n** | **m** | **f** | **iter** | **feval** | **CPU(s)** | **inercia** |
|---|---|---|---|---|---|---|---|
| bt1 | 2 | 1 | -1.00E+00 | 10 | 12 | 1.65E-01 | 1 |
| bt2 | 3 | 1 | 3.26E-02 | 10 | 13 | 4.14E-03 | 1 |
| bt4 | 3 | 2 | -1.86E+01 | 0 | 1 | 3.40E-03 | 0 |
| bt5 | 3 | 2 | 9.62E+02 | 6 | 7 | 5.98E-03 | 1 |
| bt6 | 5 | 2 | 2.77E-01 | 8 | 10 | 2.96E-03 | 1 |
| bt7 | 5 | 3 | 9.09E+02 | 0 | 1 | 7.99E-04 | 0 |
| bt8 | 5 | 2 | 3.00E+00 | 0 | 1 | 1.23E-03 | 0 |
| bt9 | 4 | 2 | -3.02E+00 | 1 | 3 | 2.03E-03 | 0 |
| bt11 | 5 | 3 | 8.25E-01 | 7 | 9 | 4.06E-03 | 1 |
| bt12 | 5 | 3 | 6.19E+00 | 3 | 4 | 9.69E-03 | 1 |
| byrdsphr | -problema mal escalado- | | | iter = | 1 | 0 | 0 |
| catena | 32 | 11 | -2.31E+04 | 7 | 10 | 5.37E-03 | 1 |
| catenary | 496 | 166 | 0.00E+00 | 0 | 1 | 1.66E-02 | 0 |
| dixchlng | 10 | 5 | 2.47E+03 | 8 | 10 | 3.14E-03 | 1 |
| dtoc1l | 14985 | 9990 | 1.25E+02 | 5 | 6 | 3.86E+00 | 1 |
| dtoc1na | 1485 | 990 | 1.27E+01 | 5 | 6 | 3.91E-01 | 1 |
| dtoc1nb | 1485 | 990 | 1.59E+01 | 5 | 6 | 3.91E-01 | 1 |
| dtoc1nc | 1485 | 990 | 2.50E+01 | 15 | 31 | 7.26E-01 | 1 |
| dtoc1nd | 735 | 490 | 1.22E+01 | 1 | 5 | 1.33E-01 | 0 |
| dtoc2 | 5994 | 3996 | 5.09E-01 | 7 | 14 | 9.95E-01 | 1 |
| dtoc4 | 14996 | 9997 | 2.87E+00 | 3 | 4 | 2.64E+00 | 1 |
| dtoc5 | 9998 | 4999 | 1.54E+00 | 3 | 4 | 7.03E-01 | 1 |
| dtoc6 | 10000 | 5000 | 1.35E+05 | 21 | 44 | 3.52E+00 | 1 |
| eigena2 | 110 | 55 | 2.85E+02 | 0 | 1 | 7.37E-03 | 0 |
| eigenaco | 110 | 55 | 2.85E+02 | 0 | 1 | 6.21E-03 | 0 |
| eigenb2 | 110 | 55 | 6.58E+02 | 0 | 1 | 8.60E-03 | 0 |
| eigenbco | 110 | 55 | 1.90E+01 | 0 | 1 | 5.78E-03 | 0 |
| eigenc2 | 462 | 231 | 1.01E+04 | 0 | 1 | 8.33E-02 | 0 |
| eigencco | 30 | 15 | 1.90E+01 | 0 | 1 | 1.01E-02 | 0 |
| gilbert | 10 | 1 | 1.93E+02 | 0 | 1 | 1.94E-03 | 0 |
| hs006 | 2 | 1 | 0.00E+00 | 5 | 15 | 2.29E-03 | 1 |
| hs007 | 2 | 1 | -3.91E-01 | 0 | 1 | 7.72E-04 | 0 |
| hs009 | 2 | 1 | 0.00E+00 | 0 | 1 | 8.96E-04 | 0 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| hs026 | 3 | 1 | 2.82E-10 | 16 | 17 | 4.26E-03 | 1 |
| hs027 | 3 | 1 | 1.22E-02 | 1 | 2 | 1.28E-03 | 0 |
| hs039 | 4 | 2 | -3.02E+00 | 1 | 3 | 1.82E-03 | 0 |
| hs040 | 4 | 3 | -2.50E-01 | 3 | 4 | 9.68E-03 | 1 |
| hs046 | 5 | 2 | 4.57E-08 | 8 | 9 | 2.34E-03 | 1 |
| hs047 | 5 | 3 | 1.58E+00 | 3 | 5 | 1.08E-02 | 0 |
| hs049 | 5 | 2 | 4.57E-06 | 11 | 12 | 3.31E-03 | 1 |
| hs050 | 5 | 3 | 6.38E-13 | 8 | 9 | 2.74E-03 | 1 |
| hs061 | 3 | 2 | 0.00E+00 | 0 | 1 | 4.67E-03 | 0 |
| hs077 | 5 | 2 | 2.42E-01 | 8 | 10 | 2.91E-03 | 1 |
| hs078 | 5 | 3 | -2.92E+00 | 3 | 4 | 1.71E-03 | 1 |
| hs079 | 5 | 3 | 7.88E-02 | 4 | 5 | 1.88E-03 | 1 |
| hs100lnp | 7 | 2 | 7.14E+02 | 0 | 1 | 9.22E-04 | 0 |
| hs111lnp | 10 | 3 | -2.10E+01 | 0 | 1 | 1.07E-03 | 0 |
| lch | 600 | 1 | 2.59E+05 | 0 | 1 | 1.53E-02 | 0 |
| maratos | 2 | 1 | -1.00E+00 | 11 | 13 | 3.14E-03 | 1 |
| maratos2 | 2 | 1 | -1.00E+00 | 11 | 14 | 3.28E-03 | 1 |
| mwright | 5 | 3 | 5.02E+01 | 1 | 2 | 1.43E-03 | 0 |
| orthrdm2 | 4003 | 2000 | 1.56E+02 | 6 | 9 | 7.96E-01 | 1 |
| orthrds2 | 203 | 100 | 3.08E+01 | 4 | 7 | 2.03E-02 | 0 |
| orthrega | 517 | 256 | 5.92E+02 | 2 | 4 | 2.96E-02 | 0 |
| orthregb | 27 | 6 | 0.00E+00 | 0 | 1 | 1.08E-03 | 0 |
| orthregc | 10005 | 5000 | 1.91E+02 | 2 | 5 | 4.78E+00 | 0 |
| orthregd | 10003 | 5000 | 1.52E+03 | 7 | 11 | 5.08E+00 | 1 |
| orthrgdm | 10003 | 5000 | 1.51E+03 | 8 | 13 | 5.52E+00 | 1 |
| orthrgds | 10003 | 5000 | 7.62E+01 | 0 | 1 | 3.77E+00 | 0 |
| | | | | 248 | 391 | 33.7852 | 30 |

**Conclusiones**

1. Nuestro método PCS resuelve eficientemente problemas de diferentes tamaños y números de restricciones, sin embargo, de los 59 problemas solamente 30 tuvieron la inercia necesaria para su resolución a través del método lo cuál dejó fuera a casi la mitad de problemas.

2. El tiempo de resolución en 52 de los problemas es menor a un segundo, de los 30 problemas con inercia correcta sólo 4 se tardaron más de un segundo en resolverse siendo 5.52 segundos el tiempo máximo de terminación del método en cualquier problema.

3. Solamente en un problema no se pudo encontrar solución por mal escalamiento, el cual apareció mal escalado (*byrdsphr*) desde la primera iteración del método.

4. La inercia de los problemas se parece más entre familias de problemas que entre número de variables o de restricciones, por ejemplo, toda la familia de problemas *eigen* tiene

inercia incorrecta a partir de la primera iteración aunque ninguno tenga más de 1000 variables ó 300 restricciones, sin embargo la familia de problemas *dtoc*, la cual llega a alcanzar las 14996 variables con 9997 restricciones, tiene siempre inercia correcta y llega a una solución en un número muy chico de iteraciones.

5. El "parentezco" entre familias de problemas también se nota en número de iteraciones necesarias para la convergencia del método y en tiempo de ejecución del método, es decir, la eficiencia y eficacia del método depende mucho más del tipo de problema que del número de variables o restricciones.

6. Buscar una solución a no tener la inercia correcta podría ser el siguiente paso a seguir (tal vez usar gradiente conjugado proyectado, regiones de confianza o métodos cuasi-Newton).

7. En la sección de **Anexos** se habla un poco del comportamiento de las $\mu'$s y de las $\alpha'$s en cada problema y cómo pueden interactuar con el progreso del algoritmo paso a paso.

1. $maxiter = 100$ , $TOL = 10^{-7}$

2. Condiciones de paro

$$\begin{aligned} ||c(x_k)||_2 \leq &\quad TOL \\ ||\nabla \text{Ł}(x_k, \lambda_k)||_2 \leq &\quad TOL \\ iter \geq &\quad maxiter \end{aligned}$$

3. Reporte del experimento

Cuadro 2: **PCS local** (*sin función de mérito*)

| **problema** | **n** | **m** | **f** | **iter** | **feval** | **CPU(s)** | **inercia** |
|---|---|---|---|---|---|---|---|
| bt1 | 2 | 1 | -1.00E+00 | 10 | 11 | 1.97E-01 | 1 |
| bt2 | 3 | 1 | 3.26E-02 | 11 | 12 | 2.84E-03 | 1 |
| bt4 | 3 | 2 | -1.86E+01 | 0 | 1 | 1.77E-03 | 0 |
| bt5 | 3 | 2 | 9.62E+02 | 6 | 7 | 2.35E-03 | 1 |
| bt6 | 5 | 2 | 2.77E-01 | 12 | 13 | 3.12E-03 | 1 |
| bt7 | 5 | 3 | 9.09E+02 | 0 | 1 | 5.72E-04 | 0 |
| bt8 | 5 | 2 | 3.00E+00 | 0 | 1 | 6.41E-04 | 0 |
| bt9 | 4 | 2 | -4.03E+00 | 1 | 2 | 9.28E-04 | 0 |
| bt11 | 5 | 3 | 8.25E-01 | 7 | 8 | 1.98E-03 | 1 |
| bt12 | 5 | 3 | 6.19E+00 | 3 | 4 | 1.20E-03 | 1 |
| byrdsphr | 3 | 2 | -2.26E+12 | 1 | 2 | 8.61E-04 | 0 |
| catena | 32 | 11 | -2.31E+04 | 6 | 7 | 2.23E-03 | 1 |
| catenary | 496 | 166 | 0.00E+00 | 0 | 1 | 6.90E-03 | 0 |
| dixchlng | 10 | 5 | 2.47E+03 | 10 | 11 | 2.93E-03 | 1 |
| dtoc1l | 14985 | 9990 | 1.25E+02 | 6 | 7 | 4.40E+00 | 1 |
| dtoc1na | 1485 | 990 | 1.27E+01 | 5 | 6 | 3.36E-01 | 1 |
| dtoc1nb | 1485 | 990 | 1.59E+01 | 5 | 6 | 3.42E-01 | 1 |
| dtoc1nc | 1485 | 990 | 2.49E+01 | 3 | 4 | 2.71E-01 | 0 |
| dtoc1nd | 735 | 490 | 2.85E+03 | 1 | 2 | 1.45E-01 | 0 |
| dtoc2 | 5994 | 3996 | 8.62E-01 | 1 | 2 | 3.06E-01 | 0 |
| dtoc4 | 14996 | 9997 | 2.87E+00 | 3 | 4 | 2.43E+00 | 1 |
| dtoc5 | 9998 | 4999 | 1.54E+00 | 3 | 4 | 6.53E-01 | 1 |
| dtoc6 | 10000 | 5000 | 1.35E+05 | 11 | 12 | 1.95E+00 | 1 |
| eigena2 | 110 | 55 | 2.85E+02 | 0 | 1 | 7.52E-03 | 0 |
| eigenaco | 110 | 55 | 2.85E+02 | 0 | 1 | 7.31E-03 | 0 |
| eigenb2 | 110 | 55 | 6.58E+02 | 0 | 1 | 4.97E-03 | 0 |
| eigenbco | 110 | 55 | 1.90E+01 | 0 | 1 | 5.31E-03 | 0 |
| eigenc2 | 462 | 231 | 1.01E+04 | 0 | 1 | 7.34E-02 | 0 |
| eigencco | 30 | 15 | 1.90E+01 | 0 | 1 | 1.48E-03 | 0 |
| gilbert | 10 | 1 | 1.93E+02 | 0 | 1 | 1.01E-03 | 0 |
| hs006 | 2 | 1 | 0.00E+00 | 3 | 4 | 1.14E-03 | 1 |
| hs007 | 2 | 1 | -3.91E-01 | 0 | 1 | 5.39E-04 | 0 |
| hs009 | 2 | 1 | 0.00E+00 | 0 | 1 | 5.10E-04 | 0 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| hs026 | 3 | 1 | 2.82E-10 | 16 | 17 | 3.12E-03 | 1 |
| hs027 | 3 | 1 | 1.22E-02 | 1 | 2 | 9.01E-04 | 0 |
| hs039 | 4 | 2 | -4.03E+00 | 1 | 2 | 8.40E-04 | 0 |
| hs040 | 4 | 3 | -2.50E-01 | 3 | 4 | 1.43E-03 | 1 |
| hs046 | 5 | 2 | 3.68E-10 | 11 | 12 | 3.22E-03 | 1 |
| hs047 | 5 | 3 | 8.50E-10 | 15 | 16 | 3.20E-03 | 1 |
| hs049 | 5 | 2 | 6.96E-09 | 15 | 16 | 3.18E-03 | 1 |
| hs050 | 5 | 3 | 6.38E-13 | 8 | 9 | 1.93E-03 | 1 |
| hs061 | 3 | 2 | 0.00E+00 | 0 | 1 | 3.50E-03 | 0 |
| hs077 | 5 | 2 | 2.42E-01 | 11 | 12 | 2.44E-03 | 1 |
| hs078 | 5 | 3 | -2.92E+00 | 4 | 5 | 1.42E-03 | 1 |
| hs079 | 5 | 3 | 7.88E-02 | 4 | 5 | 1.38E-03 | 1 |
| hs100lnp | 7 | 2 | 7.14E+02 | 0 | 1 | 7.51E-04 | 0 |
| hs111lnp | 10 | 3 | -2.10E+01 | 0 | 1 | 7.73E-04 | 0 |
| lch | 600 | 1 | 2.59E+05 | 0 | 1 | 6.14E-03 | 0 |
| maratos | 2 | 1 | -1.00E+00 | 11 | 12 | 2.49E-03 | 1 |
| maratos2 | 2 | 1 | -1.00E+00 | 11 | 12 | 3.63E-03 | 1 |
| mwright | 5 | 3 | 5.02E+01 | 1 | 2 | 9.42E-04 | 0 |
| orthrdm2 | 4003 | 2000 | 5.28E+02 | 3 | 4 | 5.52E-01 | 0 |
| orthrds2 | 203 | 100 | 3.58E+02 | 3 | 4 | 7.99E-03 | 0 |
| orthrega | 517 | 256 | 2.33E+03 | 2 | 3 | 1.89E-02 | 0 |
| orthregb | 27 | 6 | 0.00E+00 | 0 | 1 | 1.36E-03 | 0 |
| orthregc | 10005 | 5000 | 3.41E+02 | 2 | 3 | 5.03E+00 | 0 |
| orthregd | 10003 | 5000 | 8.70E+03 | 3 | 4 | 4.34E+00 | 0 |
| orthrgdm | 10003 | 5000 | 6.01E+03 | 2 | 3 | 4.16E+00 | 0 |
| orthrgds | 10003 | 5000 | 7.62E+01 | 0 | 1 | 3.89E+00 | 0 |
| | | | | 162 | 206 | 2.92E+01 | 26 |

1. En esta segunda tabla se muestran los resultados de un método similar al anterior pero que no incluye función de mérito en su programación. Es una simplificación al programa $PCS_{global}$ que no converge globalmente aunque el problema sí tenga solución en el programa anterior.

2. De los 30 problemas que el programa $PCS_{global}$ logró resolver, 25 problemas siguieron siendo solucionables por el nuevo programa $PCS_{local}$ ( *dtoc1nc, dtoc2, orthrdm2, orthregd, orthrgdm* fueron los 5 problemas que ya no se lograron solucionar).

3. Solamente hubo un problema que no se puede resolver con $PCS_{global}$ que $PCS_{local}$ sí pudo resolver (*hs047*). Esto puede que se deba a que el programa $PCS_{global}$, al acortar el tamaño del paso de descenso con $\alpha$, pudo haberse quedado " atorado " en una zona con inercia incorrecta; mientras tanto, $PCS_{local}$ no acortó ningún paso y se "saltó" la zona de inercia incorrecta.

4. El tiempo máximo que se tardó $PCS_{local}$ en resolver un problema fueron 4.40 segundos y el mayor número de evaluaciones de la función objetivo fue 17.

5. La familia de problemas $hs0\cdots$ solucionables fue la que más evaluaciones de la función, su gradiente y hessiana necesitó en general.

6. Para poder comparar de manera más práctica los dos algoritmos se utilizaran perfiles de rendimiento que aparecen en la sección de **Anexos**.

7. Faltaría poner condiciones de paro similares entre los dos métodos para poder terminar de compararlos del todo ya que puede ser que la dimensionalidad de los problemas esté afectado los tiempos de paro en $PCS_{local}$ debido a que la norma 2 es más sensible a cambiar con la dimensión que la norma infinito, sin embargo, como $\mathbb{R}^n$ es un espacio completo, las normas son equivalentes y los resultados no deberían cambiar demasiado.

8. En la sección de **Anexos** se discute un poco más del efecto de la dimensión en las condiciones de paro.

9. Vale la pena notar que, por cómo está construido el algoritmo $PCS_{local}$, el número de evaluaciones de la función objetivo probablemente sea menor que en $PCS_{global}$ dado que el nuevo programa no evalua la función cada que hace un corte con $\alpha$ (no usa cortes de ese tipo $PCS_{local}$ a diferencia de $PCS_{global}$. Si evaluar la función lleva un costo muy alto para algún problema en particular, $PCS_{local}$ podría ser una mejor propuesta que $PCS_{global}$.

10. El problema de tener inercia incorrecta afecta a más de la mitad de problemas usando este algoritmo. Encontrar una solución a esto parece ser cada vez más importante.

**Anexos**

| Nombre del problema | dtoc6 |
|---|---|
| Numero de variables | 10000 |
| Numero de restricciones | 5000 |
| Numero maximo de iteraciones | 50 |
| Tolerancia | 1.00E-05 |

| | |
|---|---|
| Objetivo en el punto inicial | 2.50E+03 |
| Norma de las restricciones en el punto inicial | 1.00E+00 |
| Norma del gradiente de la Lagrangiana en el punto inicial | 7.07E+01 |

| k | f | $\|\|c\|\|$ | $\|\|gL\|\|$ | $\alpha$ | $\mu$ |
|---|---|---|---|---|---|
| 1 | 1.54E+03 | 6.16E-01 | 6.07E-01 | 5.00E-01 | 1.00E+00 |
| 2 | 2.84E+03 | 3.71E-01 | 3.69E-01 | 5.00E-01 | 9.02E+03 |
| 3 | 5.75E+03 | 2.26E-01 | 4.63E-01 | 5.00E-01 | 2.67E+04 |
| 4 | 1.00E+04 | 1.40E-01 | 6.09E-01 | 5.00E-01 | 6.12E+04 |
| 5 | 1.56E+04 | 8.54E-02 | 7.45E-01 | 5.00E-01 | 1.26E+05 |
| 6 | 2.25E+04 | 5.16E-02 | 8.73E-01 | 5.00E-01 | 2.51E+05 |
| 7 | 3.05E+04 | 3.12E-02 | 9.97E-01 | 5.00E-01 | 4.87E+05 |
| 8 | 3.98E+04 | 1.88E-02 | 1.12E+00 | 5.00E-01 | 9.23E+05 |
| 9 | 5.02E+04 | 1.14E-02 | 1.24E+00 | 5.00E-01 | 1.71E+06 |
| 10 | 6.17E+04 | 6.86E-03 | 1.35E+00 | 5.00E-01 | 3.11E+06 |
| 11 | 7.41E+04 | 4.15E-03 | 1.47E+00 | 5.00E-01 | 5.50E+06 |
| 12 | 8.70E+04 | 2.51E-03 | 1.58E+00 | 5.00E-01 | 9.40E+06 |
| 13 | 1.00E+05 | 1.51E-03 | 1.66E+00 | 5.00E-01 | 1.52E+07 |
| 14 | 1.12E+05 | 8.57E-04 | 1.52E+00 | 5.00E-01 | 2.22E+07 |
| 15 | 1.21E+05 | 5.12E-04 | 1.07E+00 | 5.00E-01 | 2.84E+07 |
| 16 | 1.24E+05 | 4.03E-04 | 8.23E-01 | 2.50E-01 | 2.89E+07 |
| 17 | 1.27E+05 | 3.14E-04 | 6.29E-01 | 2.50E-01 | 2.89E+07 |
| 18 | 1.29E+05 | 2.43E-04 | 4.79E-01 | 2.50E-01 | 2.89E+07 |
| 19 | 1.32E+05 | 1.39E-04 | 2.54E-01 | 5.00E-01 | 2.89E+07 |
| 20 | 1.35E+05 | 1.91E-05 | 1.42E-02 | 1.00E+00 | 2.89E+07 |
| 21 | 1.35E+05 | 5.07E-09 | 7.86E-06 | 1.00E+00 | 2.89E+07 |

| | |
|---|---|
| tiempo del programa | 5.20E-01 |

La tabla de arriba muestra un gran ejemplo de cómo funciona el algoritmo. En las primeras iteraciones el valor de $\mu$ es pequeo y el programa avanza minimizando $f(x)$ pero conforme se va avanzando la penalización se va incrementando hasta el punto que entre la $7^a$ y la $15^{ava}$ iteración se pierde en minimizar la norma de el gradiente de la función lagrangiana para poder ir descendiendo en la norma de las restricciones. Ya que el algoritmo pasa esa "fase"de descenso en las restricciones

y alcanza una vecindad de la solución, el parámetro $\alpha$ se empieza a relajar y el problema parece converger cuadráticamente.

Cuadro 3: $PCS_{local}$

| Nombre del problema | orthrgdm | |
|---|---|---|
| Numero de variables | 10003 | |
| Numero de restricciones | 5000 | |
| Numero maximo de iteraciones | 50 | |
| Tolerancia | 1.00E-05 | |
| | | |
| Objetivo en el punto inicial | | 0.00E+00 |
| Norma de las restricciones en el punto inicial | | 7.15E+02 |
| Norma del gradiente de la Lagrangiana en el punto inicial | | 0.00E+00 |

| k | f | $\|\|c\|\|$ | $\|\|gL\|\|$ |
|---|---|---|---|
| 1 | 5.98E+02 | 1.47E+04 | 3.91E+03 |
| 2 | 6.01E+03 | 5.15E+03 | 3.45E+04 |

| | |
|---|---|
| inercia incorrecta | |
| tiempo del programa | 1.60E-01 |

Cuadro 4: $PCS_{global}$

| k | f | $\|\|c\|\|$ | $\|\|gL\|\|$ | $\alpha$ | $\mu$ |
|---|---|---|---|---|---|
| 1 | 1.50E+02 | 3.23E+02 | 4.24E+02 | 5.00E-01 | 9.30E-01 |
| 2 | 6.02E+02 | 1.82E+02 | 5.68E+02 | 5.00E-01 | 3.72E+00 |
| 3 | 9.23E+02 | 9.99E+01 | 3.74E+02 | 5.00E-01 | 4.72E+00 |
| 4 | 1.16E+03 | 5.37E+01 | 2.47E+02 | 5.00E-01 | 6.06E+00 |
| 5 | 1.49E+03 | 5.76E+00 | 1.24E+02 | 1.00E+00 | 7.06E+00 |
| 6 | 1.51E+03 | 1.15E-01 | 4.93E+00 | 1.00E+00 | 7.06E+00 |
| 7 | 1.51E+03 | 3.94E-05 | 1.66E-03 | 1.00E+00 | 7.06E+00 |
| 8 | 1.51E+03 | 3.69E-12 | 2.33E-10 | 1.00E+00 | 7.06E+00 |

| | |
|---|---|
| tiempo del programa | 5.20E-01 |

*Orthrgdm* es un problema que encuentra solución con $PCS_{global}$ pero no con $PCS_{local}$. La $\alpha$ del problema global empieza recortando la dirección de descenso que el problema tomaría normalmen-

te, evitando que la minimización del problema dirija al algoritmo lejos de la solución, de hecho, la f va creciendo en los primeros pasos de los dos algoritmos, la diferencia es que en $PCS_{global}$ la norma de las restricciones va descenciendo. Conforme la respuesta de cada paso se acerca más y más a la factibilidad, la norma de las restricciones y del gradiente de la función lagrangiana parece descender de manera cuadrática mientras que el parámetro $\alpha$, de nuevo, se vuelve " inactivo ". La pérdida en tiempo de ejecución bien vale que el programa $PCS_{global}$ llegue a una respuesta.

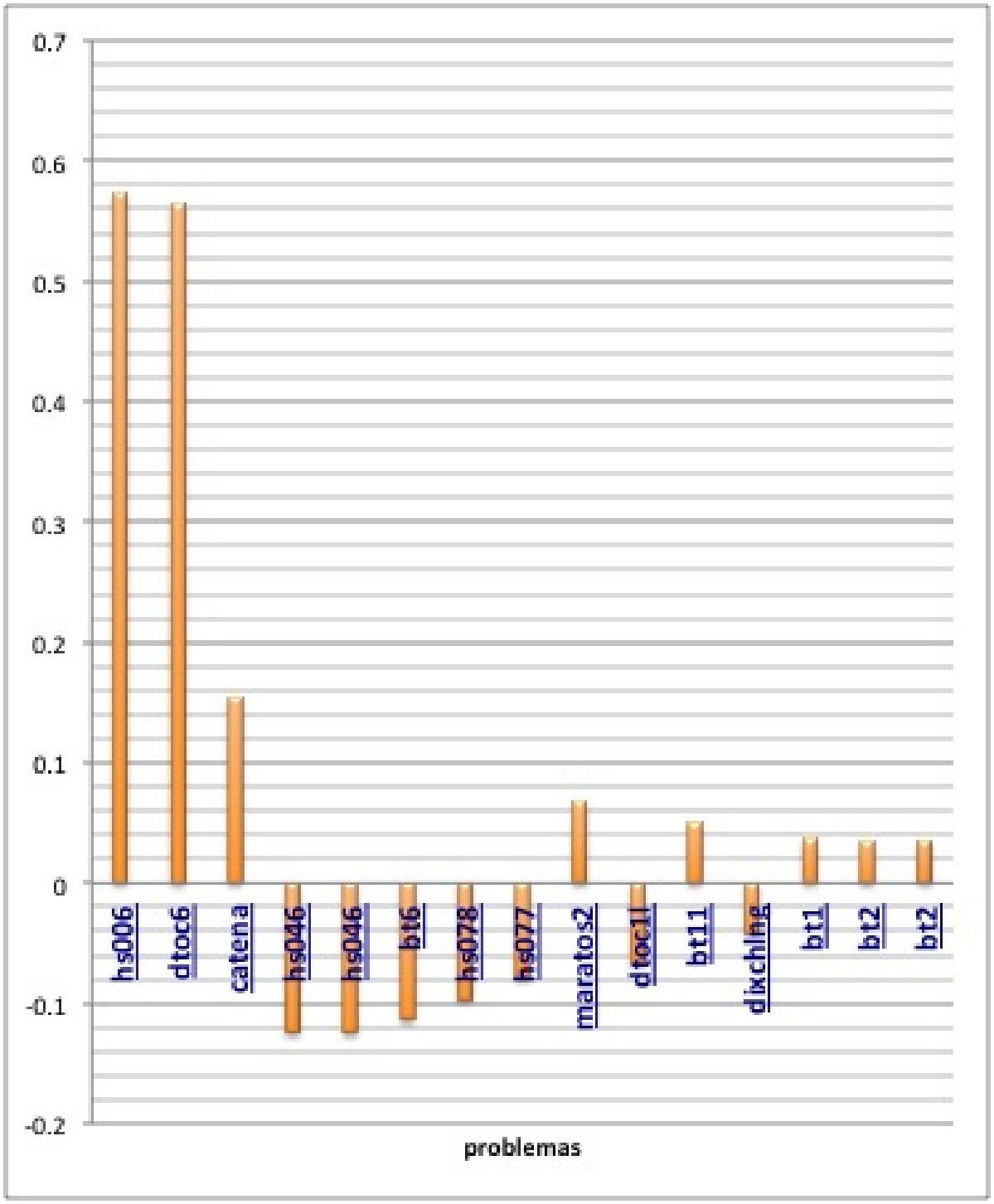
Cuadro 5: Diferencia en tiempos de ejecución de los problemas ($PCS_{global} - PCS_{local}$)

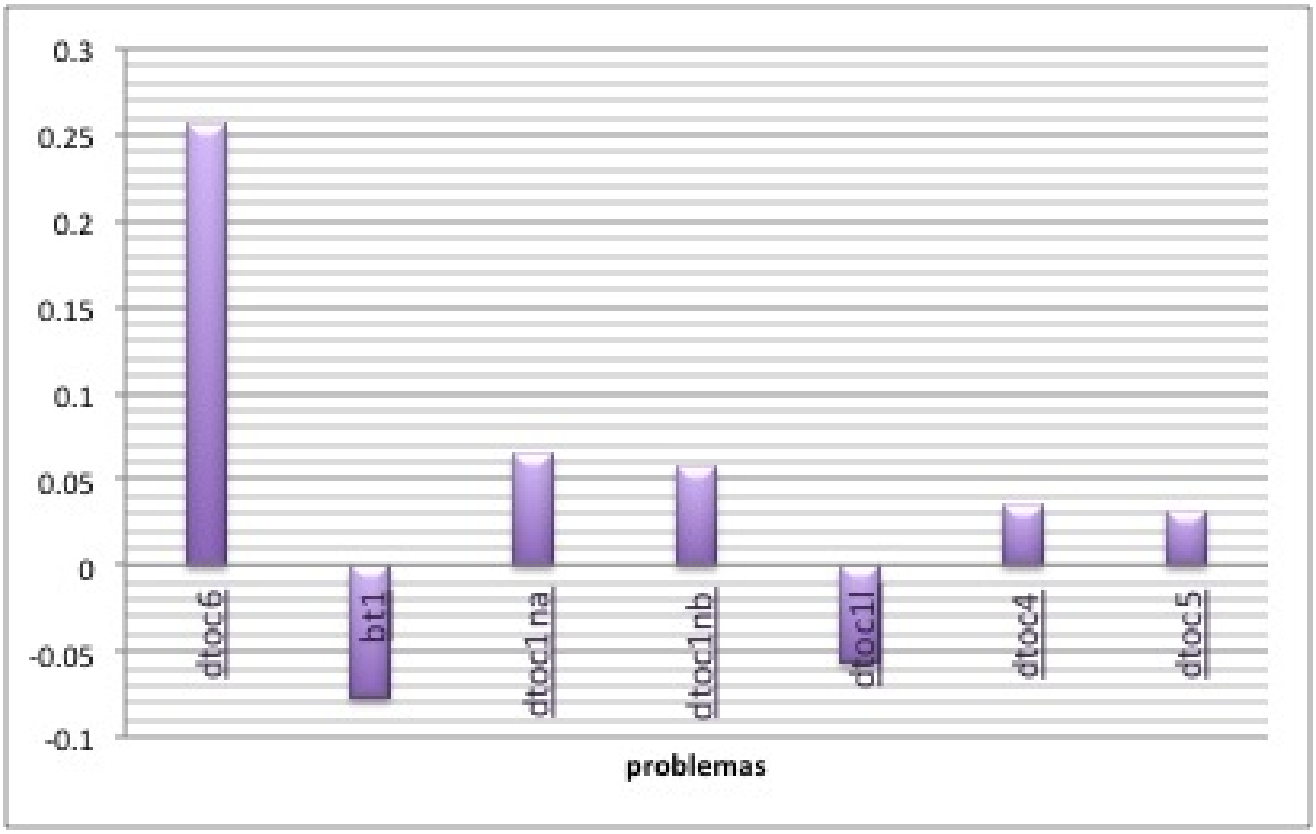
| **problema** | **dif CPU(s)** | **<1s** | **<.5s** | **<.25s** | **<.125s** | **<.1s** |
|---|---|---|---|---|---|---|
| bt1 | -3.21E-02 | 0 | 0 | 0 | 0 | 0 |
| bt2 | 1.30E-03 | 0 | 0 | 0 | 0 | 0 |
| bt4 | 1.63E-03 | 0 | 0 | 0 | 0 | 0 |
| bt5 | 3.63E-03 | 0 | 0 | 0 | 0 | 0 |
| bt6 | -1.63E-04 | 0 | 0 | 0 | 0 | 0 |
| bt7 | 2.27E-04 | 0 | 0 | 0 | 0 | 0 |
| bt8 | 5.91E-04 | 0 | 0 | 0 | 0 | 0 |
| bt9 | 1.10E-03 | 0 | 0 | 0 | 0 | 0 |
| bt11 | 2.08E-03 | 0 | 0 | 0 | 0 | 0 |
| bt12 | 8.49E-03 | 0 | 0 | 0 | 0 | 0 |
| byrdsphr | -8.61E-04 | 0 | 0 | 0 | 0 | 0 |
| catena | 3.14E-03 | 0 | 0 | 0 | 0 | 0 |
| catenary | 9.73E-03 | 0 | 0 | 0 | 0 | 0 |
| dixchlng | 2.06E-04 | 0 | 0 | 0 | 0 | 0 |
| dtoc1l | -5.39E-01 | 0 | 1 | 1 | 1 | 1 |
| dtoc1na | 5.46E-02 | 0 | 0 | 0 | 0 | 0 |
| dtoc1nb | 4.90E-02 | 0 | 0 | 0 | 0 | 0 |
| dtoc1nc | 4.55E-01 | 0 | 0 | 1 | 1 | 1 |
| dtoc1nd | -1.21E-02 | 0 | 0 | 0 | 0 | 0 |
| dtoc2 | 6.89E-01 | 0 | 1 | 1 | 1 | 1 |
| dtoc4 | 2.01E-01 | 0 | 0 | 0 | 1 | 1 |
| dtoc5 | 5.00E-02 | 0 | 0 | 0 | 0 | 0 |
| dtoc6 | 1.57E+00 | 1 | 1 | 1 | 1 | 1 |
| eigena2 | -1.45E-04 | 0 | 0 | 0 | 0 | 0 |
| eigenaco | -1.10E-03 | 0 | 0 | 0 | 0 | 0 |
| eigenb2 | 3.62E-03 | 0 | 0 | 0 | 0 | 0 |
| eigenbco | 4.73E-04 | 0 | 0 | 0 | 0 | 0 |
| eigenc2 | 9.84E-03 | 0 | 0 | 0 | 0 | 0 |
| eigencco | 8.65E-03 | 0 | 0 | 0 | 0 | 0 |
| gilbert | 9.26E-04 | 0 | 0 | 0 | 0 | 0 |
| hs006 | 1.15E-03 | 0 | 0 | 0 | 0 | 0 |
| hs007 | 2.33E-04 | 0 | 0 | 0 | 0 | 0 |

| | | | | | | |
|---|---|---|---|---|---|---|
| hs009 | 3.86E-04 | 0 | 0 | 0 | 0 | 0 |
| hs026 | 1.14E-03 | 0 | 0 | 0 | 0 | 0 |
| hs027 | 3.77E-04 | 0 | 0 | 0 | 0 | 0 |
| hs039 | 9.82E-04 | 0 | 0 | 0 | 0 | 0 |
| hs040 | 8.25E-03 | 0 | 0 | 0 | 0 | 0 |
| hs046 | -8.76E-04 | 0 | 0 | 0 | 0 | 0 |
| hs047 | 7.65E-03 | 0 | 0 | 0 | 0 | 0 |
| hs049 | 1.30E-04 | 0 | 0 | 0 | 0 | 0 |
| hs050 | 8.16E-04 | 0 | 0 | 0 | 0 | 0 |
| hs061 | 1.17E-03 | 0 | 0 | 0 | 0 | 0 |
| hs077 | 4.67E-04 | 0 | 0 | 0 | 0 | 0 |
| hs078 | 2.94E-04 | 0 | 0 | 0 | 0 | 0 |
| hs079 | 5.02E-04 | 0 | 0 | 0 | 0 | 0 |
| hs100lnp | 1.71E-04 | 0 | 0 | 0 | 0 | 0 |
| hs111lnp | 2.97E-04 | 0 | 0 | 0 | 0 | 0 |
| lch | 9.18E-03 | 0 | 0 | 0 | 0 | 0 |
| maratos | 6.49E-04 | 0 | 0 | 0 | 0 | 0 |
| maratos2 | -3.57E-04 | 0 | 0 | 0 | 0 | 0 |
| mwright | 4.89E-04 | 0 | 0 | 0 | 0 | 0 |
| orthrdm2 | 2.44E-01 | 0 | 0 | 0 | 1 | 1 |
| orthrds2 | 1.23E-02 | 0 | 0 | 0 | 0 | 0 |
| orthrega | 1.08E-02 | 0 | 0 | 0 | 0 | 0 |
| orthregb | -2.79E-04 | 0 | 0 | 0 | 0 | 0 |
| orthregc | -2.49E-01 | 0 | 0 | 0 | 1 | 1 |
| orthregd | 7.35E-01 | 0 | 1 | 1 | 1 | 1 |
| orthrgdm | 1.36E+00 | 1 | 1 | 1 | 1 | 1 |
| orthrgds | -1.20E-01 | 0 | 0 | 0 | 0 | 1 |
| | 4.56E+00 | 2 | 5 | 6 | 9 | 10 |

Arriba se encuentran las diferencias en tiempo entre usar el programa $PCS_{global}$ y el programa $PCS_{local}$. Ninguna de las diferencias parece ser significativamente grande. Es importante aclarar que la diferencia en tiempos está calculada aunque alguno de los dos algoritmos no haya podido resolver un problema (o los dos en los casos de inercia incorrecta). De las 2 únicas diferencias mayores a 1 segundo, una fue con un problema que sólo $PCS_{global}$ pudo resolver, por lo tanto no sería realmente comparable.

Abajo se encuentran en gráficas las comparaciones de rendimiento de sólo los problemas que los dos pudieron resolver y cuya diferencia en tiempo fue mayor a 1 segundo, sin embargo, como son tan pocos parece ser que la comparación no es muy informativa. Lo que se puede entender de esto es que, aunque existiese alguna diferencia en número de evaluaciones de la función o en tiempo de CPU, ésta es insignificante y la ganancia en robustez (número de problemas que $PCS_{global} resuelve$) es mucho mayor que su pérdida en " eficiencia ". De hecho, al comparar la suma de diferencias (cuánto más se tarda $PCS_{global}$ en todos los problemas juntos), sólo se pierden 4.56 segundos, y si descontamos los problemas que $PCS_{local}$ no puede resolver, esta diferencia baja a menos de 2 segundos en total.


0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
hs006
dtoc6
catena
hs046
hs046
bt6
hs078
hs077
marato2
dtoc11
bt11
dixchlm8
bt1
bt2
bt2
problemas




La gráfica de arriba representa la comparación de los problemas que los dos algoritmos lograron resolver en término de número de evaluaciones de la función. Como las primeras 3 barras dominan a las demás en magnitud, se puede decir que el programa $PCS_{global}$ evalua más veces la función que $PCS_{local}$

La gráfica de abajo representa la comparación de los problemas en término de tiempo de ejecución. Dtoc6 es el único problema que significativamente tiene una diferencia en las dos comparaciones, agregarle una penalización a este problema no es muy efectivo en general.