Curso de Programación en C++.
TPLR-2015. Recuperatorio de Trabajo Práctico de Laboratorio. [2015-09-17]

Ejercicios

[Ej. 1] **[contains]** Escribir una función
`bool contains(list<int> &L1,list<int> &L2);`
que determina si la lista `L1` contiene a la lista `L2`. Es decir cada elemento de `L1` debe estar en `L2`, teniendo en cuenta la cantidad de veces que ocurre. Por ejemplo si `L1=(3,2,1,2,3)`, entonces

- Para `L2=(3,3) -> true`
- Para `L2=(1,1) -> false`

Ayuda:

- Escribir una función
  `void list2map(list<int> &L1,map<int,int> &M);`
  que deja en `M[x]` la cantidad de veces que aparece `x` en `L`. Por ejemplo:
  - `L1=(3,2,1,2,3) -> M={1->1,2->2,3->2}`
  - `L2=(3,3) -> M={3->2}`
  - `L2=(1,1) -> M={1->2}`
- Contruir los `M1` y `M2` correspondientes a `L1` y `L2`.
- Para cada elemento en `L2` chequear que la el conteo de veces que aparece en `L2` sea menor o igual que las veces que aparece en `M1`.

[Ej. 2] **[getjaco]** Consideremos funciones vectoriales de tipo

$$f : \mathbb{R}^m \to \mathbb{R}^n \tag{1}$$

donde m, n son enteros positivos. Queremos escribir una función `GetJaco()` que calcule por diferencias el gradiente de estas funciones, es decir

$$J : \mathbb{R}^m \to \mathbb{R}^{n \times m} \tag{2}$$

donde

$$J(j,k) = \frac{\partial f_j}{\partial x_k}. \tag{3}$$

La aproximación por diferencias es

$$J(:,k) = \frac{f(x + \epsilon \hat{\mathbf{e}}_k)) - f(x - \epsilon \hat{\mathbf{e}}_k))}{2\epsilon} \tag{4}$$

donde $\hat{\mathbf{e}}_k)$ es un vector con todos 0's menos un 1 en la posición k.

Las funciones f las representamos con una clase polimórfica

```
class vecfun_t {
  public:
  virtual VectorXd eval(VectorXd &x)=0;
};
```

donde **eval()** representa la evaluación de la función.

Consigna 1: Escribir una función
void GetJaco(vecfun_t &F,VectorXd &x,MatrixXd &J,double eps=1e-5); que cumple la tarea indicada, devolviendo por **J** el Jacobiano de **F** en **x**.

Consigna 2: Escribir funciones de prueba para **GetJaco()**

- **Id(x)=x** debe retornar **J=I** (matriz identidad)
- **phi(x)=||x||^2** debe retornar **J=**$\nabla(\phi) = 2x$
- Dado un vector Ω definir la función vectorial de $\mathbb{R}^3$ en $\mathbb{R}^3$: $x \to \Omega \times x$ usando la función **cross()** de Eigen. Calcular el Jacobiano con **GetJaco()** que debe retornar

$$\begin{bmatrix} 0 & -\Omega_z & \Omega_y \\ \Omega_z & 0 & -\Omega_x \\ -\Omega_y & \Omega_x & 0 \end{bmatrix} \tag{5}$$

mstorti@galileo/texstuff-1.4.0-67-ge52a7899/Tue Jun 22 09:06:33 2021 -0300