Übung 1 *Normen im $\mathbb{R}^n$*

a) Zeichne die Einheitssphäre

$$S := \{x \in \mathbb{R}^2 | \ \|x\|_p = 1\}$$

für $p = 1, 2, \infty$.

b) Es wurde in der Vorlesung gezeigt, dass im $\mathbb{R}^n$ alle Normen äquivalent sind. Berechne explizit die sechs Koeffizienten, mit welchen die Normen $\|x\|_1$, $\|x\|_2$ und $\|x\|_\infty$ (möglichst gut) gegeneinander abgeschätzt werden können. (Was kann man über $n \longrightarrow \infty$ sagen?)

(4 Punkte)

Übung 2 *Normen im unendlich-dimensionalen Vektorraum*
Betrachten wir den Raum $C^0([0,1], \mathbb{R})$ der auf dem Intervall $[0,1]$ stetigen Funktionen.
Zeige:

a) Die durch die Abbildung

$$\|f\|_\infty = \max_{x \in [0,1]} |f(x)| \qquad f \in C^0([0,1], \mathbb{R})$$

definerte Norm besitzt die Eigenschaften einer Norm.

b) Die durch die Abbildung

$$\|f\|_1 = \int_0^1 |f(x)| dx \qquad f \in C^0([0,1], \mathbb{R})$$

definerte Norm besitzt die Eigenschaften einer Norm.

c) Betrachte für $x_k = \frac{1}{k},\ k \in \mathbb{N},\ k > 0$ die Funktionenfolge

$$u_k(x) = \begin{cases} 0 & \text{für } x \in [0,1] \backslash [x_k, x_{k+1}] \\ \sin\left(\frac{x_k - x}{x_k - x_{k+1}} \cdot \pi\right) & \text{für } x \in [x_k, x_{k+1}] \end{cases}$$

und berechne $\|u_k\|_1$ und $\|u_k\|_\infty$ für $k \longrightarrow \infty$.
Warum können diese beiden Normen nicht äquivalent sein?

(5 Punkte)

Übung 3 *Frobeniusnorm*
Die Frobenius-Norm einer Matrix $A \in \mathbb{K}^{n \times n}$ ist definiert als

$$\|A\|_F = \left(\sum_{i,j=1}^{n} |a_{ij}|^2 \right)^{\frac{1}{2}}.$$

Zeige:

(i) Die Frobenius-Norm besitzt die allgemeinen Eigenschaften einer Norm.

(ii) Sie ist verträglich mit der euklidischen Vektornorm $\|\cdot\|_2$.

(iii) Sie ist submultiplikativ.

Warum ist $\|\cdot\|_F$ dennoch keine natürliche Matrixnorm? Welche Matrixnorm ist die der euklidischen Vektornorm zugeordnete natürliche Matrixnorm? **(5 Punkte)**

Übung 4 *Hilbert-Matrix und Frobenius-Norm (Praktische Übung)*

Die sogenannte Hilbert-Matrix der Ordnung $n \geq 1$ ist eine quadratische, symmetrisch positiv definite Matrix mit den Komponenten

$$h_{ij} = \frac{1}{i+j-1} \qquad (1 \leq i,j \leq n) \tag{1}$$

(Näheres unter `http://de.wikipedia.org/wiki/Hilbert-Matrix`)

Die Inverse der Hilbert-Matrix hat die Komponenten:

$$(H_n^{-1})_{ij} = \frac{(-1)^{i+j}}{i+j-1} \cdot \frac{(n+i-1)!(n+j-1)!}{((i-1)!(j-1)!)^2(n-i)!(n-j)!} \tag{2}$$

Gegeben ist das folgende Gerüst für ein C++-Programm namens `hilbertmatrix`, welches

(a) die Hilbert-Matrix der Ordnung $n = 5$, also H_5, dessen Inverse H_5^{-1} und die beiden Matrizenprodukte $H_5 \cdot H_5^{-1}$ und $H_5^{-1} \cdot H_5$ berechnen soll.

(b) für einen beliebigen Vektor $x \in \mathbb{R}^5$ ($x \neq 0$) deiner Wahl die Verträglichkeit der Frobeniusnorm mit der euklidischen Vektornorm demonstrieren soll.

Die `//TODO:...`-Abschnitte sollen gefüllt werden. Die zu implementierenden C++-Funktionen sollen keine Bildschirmausgaben machen außer im Fehlerfall. Alle Bildschirmausgaben sollen im Hauptprogramm stattfinden.

```
// g++ -I../hdnum/ -o hilbertmatrix hilbertmatrix.cc
#include <iostream>
#include <cstdlib>
#include "hdnum.hh"

// Funktionsschablone zur Berechnung einer Hilbert-Matrix der Ordnung N
template<class NumberType>
void HilbertMatrix( hdnum::DenseMatrix<NumberType> &A ){
  int M( A.rowsize() );
  int N( A.colsize() );
  if(M!=N){
    HDNUM_ERROR("Matrix must be quadratic!");
  }
  // TODO: Elemente einer Hilbert-Matrix der Ordnung N hier berechnen!
}

// Funktion zur Berechnung einer Hilbert-Matrix der Ordnung N
// Wir verwenden hier den Typ long long anstatt int, damit auch grössere
// ganze Zahlen dargestellt werden können.
long long factorial( int n ){
  long long result(1);
  int counter(1);
  // TODO: Die Fakultaet der Zahl n soll hier berechnet und der
  // Variablen result zugewiesen werden!

  return result;
}
```

```
// Funktionsschablone zur Berechnung der Inversen einer Hilbert-Matrix der Ordnung N
template<class NumberType>
void HilbertInverse( hdnum::DenseMatrix<NumberType> &A ){
  int M(A.rowsize());
  int N(A.colsize());
  if(M!=N){
    HDNUM_ERROR("Matrix must be quadratic!");
  }
  // TODO: Elemente der Inversen einer Hilbert-Matrix der Ordnung N hier berechnen!
}


// Funktionsschablone zur Berechnung der Frobenius-Norm einer Matrix
template<class NumberType>
NumberType FrobeniusNorm( const hdnum::DenseMatrix<NumberType> &A ){
  int M(A.rowsize());
  int N(A.colsize());
  if(M!=N){
    HDNUM_ERROR("Matrix must be quadratic!");
  }
  NumberType result=0.0;
  // TODO: Die Frobenius-Norm der Matrix A hier berechnen und der Variablen result zuweisen!
  return result;
}


// Hauptprogramm
int main(int argc, char ** argv){

  // Wir betrachten nur mal den Fall n=5:
  const int N( 5 );
  std::cout << "N = " << N << std::endl;

  // Da die Hilbertmatrix schlecht konditioniert ist, benoetigen wir eine
  // höhere Genauigkeit. Wir benutzen hier den Typ long double anstatt double.
  typedef long double REAL;

  hdnum::DenseMatrix<REAL> H(N,N);

  // pretty-printing einmal setzen für alle Matrizen
  H.scientific(false);
  H.width(15);

  HilbertMatrix( H );
  std::cout << "H = " << H << std::endl;
  // TODO: Definiere hier eine neue Matrix
  // und fülle sie mit der Werten der Inversen von H
  // Gebe sie am Bildschirm aus.
  // Fahre fort mit Teil (a) und (b) der Aufgabe...

  return 0;
}
```

Hinweise:

- Es wird die Numerikbibliothek *HDNUM* benötigt. Download über die Webseite der Vorlesung:

  `http://conan.iwr.uni-heidelberg.de/teaching/numerik0_ss2011/`

- Kompilieren:

  `g++ -I../hdnum/ -o hilbertmatrix hilbertmatrix.cc`

Dieser Kompilierbefehl wird funktionieren, wenn `factorial.cc` z.B. in einem Verzeichnis `Blatt3/` parallel zum Verzeichnis `hdnum/` liegen würde.

- Bitte den C++ *Style Guide* beachten!

(5 Punkte)