Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Μάθημα: Αριθμητική Ανάλυση

ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΕΡΓΑΣΙΑ
ΗΜΕΡΟΜΗΝΙΑ ΠΑΡΑΔΟΣΗΣ: 03/06/2023

Παραγοντοποιήσεις LU και QR

Ο κώδικας Python στο πρώτο κατά σειρά πλαίσιο παρακάτω περιλαμβάνει την υλοποίηση της συνάρτησης LUmine που δέχεται ως είσοδο έναν τετραγωνικό πίνακα A και εξάγει έναν κάτω τριγωνικό πίνακα L και έναν άνω τριγωνικό πίνακα U τέτοιους ώστε $A = LU$. Είστε ελεύθεροι/ες να κάνετε όποια αλλαγή θεωρείτε στον κώδικα αυτό για να απαντήσετε κάποια από τα επόμενα ερωτήματα.

Ο κώδικας στο επόμενο (δεύτερο) πλαίσιο περιέχει την συνάρτηση QRmine που αφορά την υλοποιήση της παραγοντοποίησης QR με την μέθοδο Gram-Schmidt[1] σε γλώσσα Python, αλλά δεν είναι ολοκληρωμένος. Συμπληρώστε τα κομμάτια που λείπουν (σημεία με "???") και ελέγξτε τον τελικό κώδικά σας εισάγοντας τετραγωνικούς πίνακες διαφόρων διαστάσεων. Ο ευκολότερος τρόπος για να ελέγξετε τον κώδικά σας είναι να βρείτε την διαφορά μεταξύ A και QR κάνοντας χρήση της συνάρτησης: np.linalg.norm(A - Q*R). Ο κώδικας αυτός θα σας φανεί επίσης χρήσιμος για να απαντήσετε κάποια από τα παρακάτω ερωτήματα.

```python
def LUmine(A):
  n = A.shape[0] # η διάσταση του πίνακα A
  L = np.matrix( np.identity(n) ) # δημιουργία μοναδιαίου μέρους του L
  U = A
  for j in range(0,n-1):
    for i in range(j+1,n):
        mult = A[i,j] / A[j,j]
        A[i, j+1:n] = A[i, j+1:n] - mult * A[j,j+1:n]
        U[i, j+1:n] = A[i, j+1:n]
        L[i,j] = mult
        U[i,j] = 0
  return L, U
```

```python
def QRmine(A):
    n = A.shape[0]
    Q = np.matrix( np.zeros( (n,n) ) )
    for j in range( ??? ): # Αυτός ο βρόχος διατρεχεί τις στήλες
      q = A[:,j]
      # Ο επόμενος βρόχος κάνει όλες τις προβολές
      for i in range( ??? ):
        length_of_leg = np.sum( A[:,j].T * Q[:,i])
        q = q -  ??? * ??? # Εδώ γίνεται η κάθε προβολή
      Q[:,j] = q / np.linalg.norm(q)
    R = ???
    return Q, R
```

Πίνακες Hilbert

Ένας πίνακας Hilbert είναι ένας πίνακας της μορφής $H_{ij} = 1/(i+j+1)$, όπου και οι δύο δείκτες i και j ξεκινούν απο το 0. Για παράδειγμα ο 4×4 πίνακας Hilbert είναι ο εξής:

$$\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \end{bmatrix}$$

[1] Η μέθοδος αυτή περιγράφεται σε πληθώρα πηγών στο διαδίκτυο όπως και σε βιβλία σχετικά με θέματα Αριθμητικής Ανάλυσης και Γραμμικής Άλγεβρας. Στην σελίδα του μαθήματος στο e-learning στην ενότητα "Πρόσθετη Βιβλιογραφία" θα βρείτε συνοπτικό κείμενο που περιγράφει τη μέθοδο.

Οι πίνακες Hilbert χρησιμοποιούνται συχνά για να ελεγχθούν αλγόριθμοι αριθμητικής γραμμικής άλγεβρας, αφού συχνά παρουσιάζουν "περίεργη συμπεριφορά"... κάτι που θα διαπιστώσετε σε λίγο εκτελόντας τα παρακάτω βήματα:

(α) Γράψτε κώδικα που να δημιουργεί έναν $n \times n$ πίνακα Hilbert και ονομάστε αυτόν τον πίνακα H. Δοκιμάστε τον κώδικά σας για διάφορες τιμές του n για να επικυρώσετε ότι λειτουργεί ορθά.

(β) Δημιουργήστε ένα διάνυσμα με στοιχεία ίσα με 1 χρησιμοποιώντας τον κώδικα $b = np.ones((n, 1))$ και στην συνέχεια επιλύστε το σύστημα $Hx = b$ κάνοντας χρήση παραγοντοποίησης που αναφέρεται στην προηγούμενη ενότητα.

(γ) Μεταβάλλεται την τιμή του πρώτου στοιχείου του b κατά πολύ λίγο (π.χ. προσθέστε σε αυτό τον αριθμό 10^{-15}) και ονομάστε b_{new} το διάνυσμα που προέκυψε. Επιλύστε το σύστημα $Hx_{new} = b_{new}$ και στην συνέχεια υπολογίστε τη μέγιστη απόλυτη διαφορά όπως προκύπτει από την παράκατω γραμμή κώδικα: $np.max(np.abs(x - xnew))$. Τι παρατηρείτε; Βρίσκεται το αποτέλεσμα σε συμφωνία με αυτό που θα αναμένατε;

(δ) Δημιουργήστε ένα διάγραμμα με τις τιμές του n στον οριζόντιο άξονα και τις τιμές της μέγιστης απόλυτης διαφοράς στον κατακόρυφο άξονα. Τι συμπεράσματα βγάζετε από αυτό το διάγραμμα όσον αφορά τη λύση του συστήματος $Hx = b$;

(ε) Γνωρίζετε ότι το απότελεσμα HH^{-1} θα πρέπει να είναι μοναδιαίος πίνακας. Για τις διάφορες τιμές του n, υπολογίστε το HH^{-1} (κάνοντας χρήση του $np.linalg.inv()$, ώστε να υπολογιστεί απευθείας ο αντίστροφος) και, στην συνέχεια, υπολογίστε την 2-νόρμα της διαφοράς του μοναδιαίου πίνακα ($np.identity(n)$) και του πίνακα HH^{-1} που υπολογίσατε. Τέλος, δημιουργήστε ένα διάγραμμα με τις τιμές του n στον οριζόντιο άξονα και τις τιμές της 2-νόρμας (όπως ορίστηκε πιο πριν) στον κατακόρυφο άξονα. Τι παρατηρείτε; Τι σημαίνει αυτό για την διαδικασία αντιστροφής των πίνακων Hilbert;

Πρόβλημα προσέγγισης

Ζητείται να κατασκευαστεί ένα πολυώνυμο βαθμού 4 που να προσεγγίζει βέλτιστα την συνάρτηση:

$$y = cos(4t) + 0,1\epsilon(t)$$

σε 50 ισαπέχοντα σημεία με t μεταξύ 0 και 1. Στο παρόν πρόβλημα θεωρούμε ότι η $\epsilon(t)$ είναι μια συνάρτηση που εξάγει κανονικά κατανεμημένες τιμές λευκού θορύβου. Στην Python μπορείτε να εισάγετε την y ως: $y = np.cos(4 * t) + 0.1 * np.random.randn(t.shape[0])$.

Στην αρχή δημιουργείστε τα διανύσματα t και y που θα περιέχουν τα δεδομένα/παρατηρήσεις. Στην συνέχεια να βρείτε το πολυώνυμο βαθμού 4 που προσεγγίζει βέλτιστα την παραπάνω συνάρτηση για τις δοθέντες παρατηρήσεις, κάνοντας χρήση της μεθόδου ελαχίστων τετραγώνων. Να επιλύσετε το σύστημα που προκύπτει κάνοντας χρήση της παραγοντοποίησης LU (τρόπος 1) και της παραγοντοποίησης QR (τρόπος 2). Να δώσετε το άθροισμα των τετραγωνικών σφαλμάτων που προκύπτουν από την παραπάνω προσέγγιση για κάθε τρόπο και να δώσετε ένα διάγραμμα που θα περιέχει τα δεδομένα και την καμπύλη βέλτιστης προσέγγισης στην οποία καταλήξατε.

Παραδοτέο στην ιστοσελίδα του μαθήματος στο elearning (Εργασία 2023):
Αρχείο .ipynb που θα περιέχει τον κώδικα και τις απάντησεις στα ανωτέρω ερωτήματα. Εναλλακτικά, μπορείτε να παραδώσετε ένα αρχείο με τον κώδικά σας και ένα αρχείο σε μορφή pdf με τις απαντήσεις σας.