4.3.2 Triangular Matrix-Vector Multiplication

Bookmark this page

Week 4 due Oct 24, 2023 19:42 IST

4.3.2 Triangular Matrix-Vector Multiplication

Start of transcript. Skip to the end.

Dr. Robert van de Geijn: So another thing that happens

is that often matrices have special structure.

Matrices with special structure, that we discussed last week,

were triangular matrices and symmetric matrices.

Again, computational scientists are

0:00 / 0:00 2.0x

Video

Download video file

Transcripts

Download SubRip (.srt) file

Download Text (.txt) file

Reading Assignment

0 points possible (ungraded)

Read Unit 4.3.2 of the notes. [LINK]

- [x] Done

Submit

Discussion

Hide Discussion

Topic: Week 4 / 4.3.2

Add a Post

Show all posts by recent activity

Algorithm Typo error for HW 4.3.2.1, 4.3.2.4 2

Algorithm 1

HW Homework 4.3.2.8/9 2

I implemented Trmv_ut_unb_var2 and test_Trmv_ut_unb_var2 (my test script is the same as the given ones except the equality is > if (isequal(Tr...

Any chance there's an issue with test_Trmvp_un_umb_var1.m?

Calculator

I've checked my MatLab code against the answer and it's exactly the same but I keep getting an error: Output argument "alpha_out" (and possib...

? Help!!! How to get rid of a vertical line on the new function window in Matlab? 3

? Answer script? 4
Hello, I did the homework and the result is correct, but not sure if I went through the process the question was asking. Is there an answer sheet ...

Question about one of the summation derivation steps 4
Hello, I am a little confused on how the sum from j=1 to n of j ends up equivalent to 2(n(n+1)/2). What am I missing here?

? Problem with testing Trmv_ln_unb_var2 8
Greetings, everyone! I managed to work out all previous exercises, but got some issues with 'Trmv_ln_unb_var2' (second half of 4.3.2.6). While ru...

? Testing scripts not working for Trmvp_un_unb_var(x)? 4
I am having difficulty getting the test_Trmvp_un_unb_var1 and var2 scripts to run properly. For some reason, they are only creating the Matrix A, ...

Homework 4.3.2.1

1/1 point (graded)

Algorithm: $y := \textsc{Trmvp\_un\_unb\_var1}(U, x, y)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where U_{TL} is 0×0, x_T, y_T are 0×1

while $m(U_{TL}) < m(U)$ **do**

Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$$\psi_1 := u_{10}^T x_0 + \upsilon_{11}\chi_1 + u_{12}^T x_2 + \psi_1$$

Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Algorithm: $y := \textsc{Trmvp\_un\_unb\_var2}(U, x, y)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where U_{TL} is 0×0, x_T, y_T are 0×1

while $m(U_{TL}) < m(U)$ **do**

Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$$y_0 := \chi_1 u_{01} + y_0$$
$$\psi_1 := \chi_1 \upsilon_{11} + \psi_1$$
$$y_2 := \chi_1 u_{21} + y_2$$

Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & A_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Write functions

[y_out] = Trmvp_un_unb_var1(U, x, y); and

[y_out] = Trmvp_un_unb_var2(U, x, y)

that compute $y := Ux + y$.

Calculator

Some links that will come in handy:

- Spark (alternatively, open the file `LAFF-2.0xM/Spark/index.html`)
- PictureFLAME (alternatively, open the file `LAFF-2.0xM/PictureFLAME/PictureFLAME.html`)

You may want to use the following scripts to test your implementations (these should be in your directory `LAFF-2.0xM/Programming/Week04/`):

- test_Trmvp_un_unb_var1.m
- test_Trmvp_un_unb_var2.m

☑ Done/Skip

✔

View document with most algorithms and implementations for this week.

Submit

Answers are displayed within the problem

Homework 4.3.2.2

1/1 point (graded)

Modify the following algorithms to compute $y := Lx + y$, where L is a lower triangular matrix:

Algorithm: $y := \text{TRMVP\_LN\_UNB\_VAR1}(L, x, y)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right)$,

$x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where L_{TL} is 0×0, x_T, y_T are 0×1

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$\psi_1 := l_{10}^T x_0 + \lambda_{11}\chi_1 + l_{12}^T x_2 + \psi_1$

Continue with

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline \end{array}\right)$$

Algorithm: $y := \text{TRMVP\_LN\_UNB\_VAR2}(L, x, y)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right)$,

$x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where L_{TL} is 0×0, x_T, y_T are 0×1

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$y_0 := \chi_1 l_{01} + y_0$
$\psi_1 := \chi_1 \lambda_{11} + \psi_1$
$y_2 := \chi_1 l_{21} + y_2$

Continue with

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \end{array}\right)$$

Calculator

$\left(\frac{x_B}{x_2} \right) \quad \left(\frac{y_B}{y_2} \right)$

endwhile

$\left(\frac{x_B}{x_2} \right) \quad \left(\frac{y_B}{y_2} \right)$

endwhile

(Just strike out the parts that evaluate to zero. We suggest you do this homework in conjunction with the next one.)

- [x] Done

The answer can be found in LAFF-2.0xM/Answers/Week04/. Just examine the code.

Submit

Answers are displayed within the problem

Homework 4.3.2.3

1/1 point (graded)
Write routines

- `[ y_out ] = Trmvp_ln_unb_var1( L, x, y )`
- `[ y_out ] = Trmvp_ln_unb_var2( L, x, y )`

that compute $y := Lx + y$.

Some links that will come in handy:

- Spark (alternatively, open the file `LAFF-2.0xM/Spark/index.html`)
- PictureFLAME (alternatively, open the file `LAFF-2.0xM/PictureFLAME/PictureFLAME.html`)

You may want to use the following scripts to test your implementations (these should be in your directory `LAFF-2.0xM/Programming/Week04/`):

- test_Trmvp_ln_unb_var1.m
- test_Trmvp_ln_unb_var2.m

- [x] Done/Skip

View document with most algorithms and implementations for this week.

Submit

Answers are displayed within the problem

Homework 4.3.2.4

1/1 point (graded)
Modify the following algorithms to compute $x := Ux$, where U is a upper triangular matrix. **You may not use y. You have to overwrite x without using work space**. Hint: Think carefully about the order in which elements of x are computed and overwritten. You may want to do this exercise hand in hand with the implementation in the next homework.

Algorithm: $y := \text{TRMVP\_UN\_UNB\_VAR1}(U, x, y)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline \end{array} \right)$,

Algorithm: $y := \text{TRMVP\_UN\_UNB\_VAR2}(U, x, y)$

Partition $U \rightarrow \left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline \end{array} \right)$,

Calculator

$$\left(\begin{array}{c|c} U_{BL} & U_{BR} \end{array} \right)$$

$x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array} \right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array} \right)$

where U_{TL} is 0×0, x_T, y_T are 0×1

while $m(U_{TL}) < m(U)$ **do**

Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array} \right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array} \right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array} \right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array} \right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array} \right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array} \right)$$

$$\psi_1 := u_{10}^T x_0 + \upsilon_{11}\chi_1 + u_{12}^T x_2 + \psi_1$$

Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array} \right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & A_{22} \end{array} \right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array} \right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array} \right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array} \right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array} \right)$$

endwhile

$$\left(\begin{array}{c|c} U_{BL} & U_{BR} \end{array} \right)$$

$x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array} \right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array} \right)$

where U_{TL} is 0×0, x_T, y_T are 0×1

while $m(U_{TL}) < m(U)$ **do**

Repartition

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array} \right) \rightarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & U_{22} \end{array} \right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array} \right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array} \right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array} \right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array} \right)$$

$$y_0 := \chi_1 u_{01} + y_0$$
$$\psi_1 := \chi_1 \upsilon_{11} + \psi_1$$
$$y_2 := \chi_1 u_{21} + y_2$$

Continue with

$$\left(\begin{array}{c|c} U_{TL} & U_{TR} \\ \hline U_{BL} & U_{BR} \end{array} \right) \leftarrow \left(\begin{array}{c|c|c} U_{00} & u_{01} & U_{02} \\ \hline u_{10}^T & \upsilon_{11} & u_{12}^T \\ \hline U_{20} & u_{21} & A_{22} \end{array} \right),$$

$$\left(\begin{array}{c} x_T \\ \hline x_B \end{array} \right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array} \right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array} \right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array} \right)$$

endwhile

- [x] Done

Explanation
The answer can be found in LAFF-2.0xM/Answers/Week04/. Just examine the code.

Submit

Answers are displayed within the problem

Homework 4.3.2.5

1/1 point (graded)
Write routines

- `[ x_out ] = Trmv_un_unb_var1( U, x )`
- `[ x_out ] = Trmv_un_unb_var2( U, x )`

that compute $x := Ux$.

Some links that will come in handy:

- Spark (alternatively, open the file `LAFF-2.0xM/Spark/index.html`)
- PictureFLAME (alternatively, open the file `LAFF-2.0xM/PictureFLAME/PictureFLAME.html`)

Calculator

You may want to use the following scripts to test your implementations (these should be in your directory `LAFF-2.0xM/Programming/Week04/`):

- test_Trmv_un_unb_var1.m
- test_Trmv_un_unb_var2.m

Done/Skip

View document with most algorithms and implementations for this week.

Submit

Answers are displayed within the problem

Homework 4.3.2.6

1/1 point (graded)

Modify the following algorithms to compute $x := Lx$, where L is a lower triangular matrix. **You may not use y. You have to overwrite x without using work space.** Hint: Think carefully about the order in which elements of x are computed and overwritten. This question is VERY tricky... You may want to do this exercise hand in hand with the implementation in the next homework.

Algorithm: $y := \text{TRMVP\_LN\_UNB\_VAR1}(L, x, y)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where L_{TL} is 0×0, x_T, y_T are 0×1

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$$\psi_1 := l_{10}^T x_0 + \lambda_{11}\chi_1 + l_{12}^T x_2 + \psi_1$$

Continue with

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Algorithm: $y := \text{TRMVP\_LN\_UNB\_VAR2}(L, x, y)$

Partition $L \rightarrow \left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right)$, $x \rightarrow \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right)$, $y \rightarrow \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right)$

where L_{TL} is 0×0, x_T, y_T are 0×1

while $m(L_{TL}) < m(L)$ **do**

Repartition

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \rightarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \rightarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

$$y_0 := \chi_1 l_{01} + y_0$$
$$\psi_1 := \chi_1 \lambda_{11} + \psi_1$$
$$y_2 := \chi_1 l_{21} + y_2$$

Continue with

$$\left(\begin{array}{c|c} L_{TL} & L_{TR} \\ \hline L_{BL} & L_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} L_{00} & l_{01} & L_{02} \\ \hline l_{10}^T & \lambda_{11} & l_{12}^T \\ \hline L_{20} & l_{21} & L_{22} \end{array}\right), \left(\begin{array}{c} x_T \\ \hline x_B \end{array}\right) \leftarrow \left(\begin{array}{c} x_0 \\ \hline \chi_1 \\ \hline x_2 \end{array}\right), \left(\begin{array}{c} y_T \\ \hline y_B \end{array}\right) \leftarrow \left(\begin{array}{c} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{array}\right)$$

endwhile

Calculator

- [x] Done/Skip

Explanation
The key is that you have to "march" from the bottom-right (BR) to the top-left (TL) through the matrix and from the bottom (B) to the top (T) through the vector, in order to avoid overwriting elements of `x` before you no longer need them.
In other words (suggested by one of the learners in Fall 2017): When you would march through the matrix from the top-left to the bottom-right then you would overwrite x0 every time you update chi1. Instead when marching through the matrix from the bottom-right to the top-left then you overwrite x2 every time you update chi1. However, for a lower triangular matrix the calculation with x2 is not needed and, thus, can be safely overwritten.

Submit

Answers are displayed within the problem

Homework 4.3.2.7

1/1 point (graded)
Write routines

- `[ x_out ] = Trmv_ln_unb_var1( L, x )`
- `[ x_out ] = Trmv_ln_unb_var2( L, x )`

that compute $x := Lx$.

Some links that will come in handy:

- Spark (alternatively, open the file `LAFF-2.0xM/Spark/index.html`)
- PictureFLAME (alternatively, open the file `LAFF-2.0xM/PictureFLAME/PictureFLAME.html`)

You may want to use the following scripts to test your implementations (these should be in your directory `LAFF-2.0xM/Programming/Week04/`):

- test_Trmv_ln_unb_var1.m
- test_Trmv_ln_unb_var2.m

- [x] Done/Skip

Explanation
The key is that you have to "march" from the bottom-right (BR) to the top-left (TL) through the matrix and from the bottom (B) to the top (T) through the vector, in order to avoid overwriting elements of `x` before you no longer need them.
In other words (suggested by one of the learners in Fall 2017): When you would march through the matrix from the top-left to the bottom-right then you would overwrite x0 every time you update chi1. Instead when marching through the matrix from the bottom-right to the top-left then you overwrite x2 every time you update chi1. However, for a lower triangular matrix the calculation with x2 is not needed and, thus, can be safely overwritten.
View document with most algorithms and implementations for this week.

Submit

Answers are displayed within the problem

Homework 4.3.2.8

Calculator

1/1 point (graded)
Develop algorithms for computing $y := U^T x + y$ and $y := L^T x + y$, where U and L are respectively upper triangular and lower triangular. Do not explicitly transpose matrices U and L. Write routines

- `[ y_out ] = Trmvp_ut_unb_var1( U, x , y )`
- `[ y_out ] = Trmvp_ut_unb_var2( U, x , y )`
- `[ y_out ] = Trmvp_lt_unb_var1( L, x , y )`
- `[ y_out ] = Trmvp_lt_unb_var2( L, x , y )`

that implement these algorithms.

Some links that will come in handy:

- Spark (alternatively, open the file `LAFF-2.0xM/Spark/index.html`)
- PictureFLAME (alternatively, open the file `LAFF-2.0xM/PictureFLAME/PictureFLAME.html`)

Sorry, no test scripts... You should be able to create them yourself now!

☑ Done/Skip

✔

No solutions... I got tired.

Submit

Answers are displayed within the problem

Homework 4.3.2.9

1/1 point (graded)
Develop algorithms for computing $x := U^T x$ and $x := L^T x$, where U and L are respectively upper triangular and lower triangular. Do not explicitly transpose matrices U and L. Write routines

- `[ x_out ] = Trmv_ut_unb_var1( U, x )`
- `[ x_out ] = Trmv_ut_unb_var2( U, x )`
- `[ x_out ] = Trmv_lt_unb_var1( L, x )`
- `[ x_out ] = Trmv_lt_unb_var2( L, x )`

that implement these algorithms.

Some links that will come in handy:

- Spark (alternatively, open the file `LAFF-2.0xM/Spark/index.html`)
- PictureFLAME (alternatively, open the file `LAFF-2.0xM/PictureFLAME/PictureFLAME.html`)

Sorry, no test scripts... You should be able to create them yourself now!

☑ Done/Skip

✔

No solutions... I got tired.

Submit

Calculator

Answers are displayed within the problem

Homework 4.3.2.10

1/1 point (graded)
How many flops are in the algorithm for computing $y := Lx + y$ that uses axpys. ($L \in \mathbb{R}^{n \times n}, x, y \in \mathbb{R}^n$)

Note: you will want to read the "Related Reading" before you answer this question.

- ( ) $2n + 1$
- ( ) $3n$
- (x) $n^2 + n$
- ( ) $2n^3$

✔

Explanation
Answer: For the **axpy** based algorithm, the cost is in the updates

- $\psi_1 := \lambda_{11}\chi_1 + \psi_1$ (which requires two flops) ; followed by
- $y_2 := \chi_1 l_{21} + y_2$.

Now, during the first iteration, y_2 and l_{21} and x_2 are of length $n-1$, so that that iteration requires $2(n-1)+2 = 2n$ flops. During the kth iteration (starting with $k=0$), y_2 and l_{21} are of length $(n-k-1)$ so that the cost of that iteration is $2(n-k-1)+2 = 2(n-k)$ flops. Thus, if L is an $n \times n$ matrix, then the total cost is given by

$$\sum_{k=0}^{n-1}[2(n-k)] = 2\sum_{k=0}^{n-1}(n-k) = 2(n+(n-1)+\cdots+1) = 2\sum_{k=1}^{n} k = 2(n+1)n/2.$$

flops. (Recall that we proved in the second week that $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$.)

Submit

Answers are displayed within the problem

Homework 4.3.2.11

4/4 points (graded)
As hinted at before: Implementations achieve better performance (finish faster) if one accesses data consecutively in memory. Now, most scientific computing codes store matrices in "column-major order" which means that the first column of a matrix is stored consecutively in memory, then the second column, and so forth. Now, this means that an algorithm that accesses a matrix by columns tends to be faster than an algorithm that accesses a matrix by rows. That, in turn, means that when one is presented with more than one algorithm, one should pick the algorithm that accesses the matrix by columns.

Our FLAME notation makes it easy to recognize algorithms that access the matrix by columns. For example, in this unit, if the algorithm accesses submatrix a_{10}^T or a_{12}^T, then it accesses the matrix by rows.

For each of these, which algorithm accesses the matrix by columns:

• For $y := Ux + y$, does the better algorithm use a dot or an axpy?

axpy ✔ Answer: axpy

Calculator

- For $y := Lx + y$, does the better algorithm use a dot or an axpy?

axpy ✔ Answer: axpy

- For $y := U^T x + y$, does the better algorithm use a dot or an axpy?

dot ✔ Answer: dot

- For $y := L^T x + y$, does the better algorithm use a dot or an axpy?

dot ✔ Answer: dot

Submit

Answers are displayed within the problem

< Previous Next >

edX

[About](#)
[Affiliates](#)
[edX for Business](#)
[Open edX](#)
[Careers](#)
[News](#)

Legal

[Terms of Service & Honor Code](#)
[Privacy Policy](#)
[Accessibility Policy](#)
[Trademark Policy](#)
[Sitemap](#)
[Cookie Policy](#)
Your Privacy Choices

Connect

[Idea Hub](#)

Calculator

Contact Us
Help Center
Security
Media Kit

GET IT ON Google play
Download on the App Store

Calculator