Laboratorio 2 Algoritmos Numéricos

Isaac Espinoza

14 de junio de 2019

En este informe se presentarán los resultados del Laboratorio 2 de Algoritmos Numéricos. Se muestran los resultados obtenidos al resolver matrices aplicando los métodos: LU, Cholesky, QR, Givens, Mínimos Cuadrados y Gauss Seidel. Dichos métodos fueron implementados en C++ utilizando el módulo de armadillo, además se comparará utilizando gráficos y tablas la eficiencia y la eficacia .

1. Comparación Errores: Matlab vs Armadillo

La tabla 1 compara los errores en métodos directos obtenidos en matlab y en armadillo. Estos errores se asocian a las operaciones de punto flotante. Como se puede ver, los errores son muy similares y generalmente del mismo orden, entre e^{-14} y e^{-16}, por lo tanto se puede considerar que ambas soluciones son lo suficientemente eficaces, considerando que se usó el tipo de dato **double** tanto en matlab como en C++ Armadillo. También se muestran los gráficos seminormalizados con logaritmo del valor de los errores del método Gauss Seidel en cada iteración. Cabe destacar que en ambos casos (matlab y armadillo) se usó tolerancia de 10^{-18} y un límite de 1000 iteraciones.
En el gráfico 1, ambos alcanzan la tolerancia y cerca de la iteración 80 experimentan aumentos y disminuciones del error, lo cual contrasta con la disminución constante experimentada en las iteraciones anteriores. Además, se observan más perturbaciones en el error de Armadillo.
En el gráfico 2, ambos alcanzan la tolerancia y cerca de la iteración 300 nuevamente experimentan aumentos de error y disminuciones que se diferencian del comportamiento de las iteraciones anteriores.

Finalmente, en el gráfico 3 ambos métodos no alcanzan la tolerancia deseada y además se comportan de manera muy similar, lo que provoca que la curva del error de Matlab se sobreponga a simple vista respecto a la curva de Armadillo.
Como se ve en las figuras de resultados 4, 5 y 6, a simple vista todos los puntos (que tienen marcadores distintos) se superponen entre sí, es decir, poseen valores iguales o muy similares. No se observa ningún outlier de algún método en específico.
Finalmente, se incluye la tabla 3 en la cual se comparan las iteraciones y error obtenido en el método de Gauss Seidel para Armadillo y Matlab. El número de iteraciones fue similar en ambos casos, pudiendo observar una diferencia mínima. En el caso de la matriz de 4225x4225, ambos no llegaron a converger a la tolerancia de 10^{-18}, pero los dos alcanzaron errores muy similares.

2. Comparación Eficiencia: Matlab vs Armadillo

En esta sección se compararán los tiempos obtenidos al usar los métodos en Armadillo vs Matlab.
En la figura 7, se tienen dos gráficos que muestran los tiempos para los métodos de LU, Cholesky, QR, Mínimos Cuadrados y otro gráfico en que solo se muestra Givens y Seidel, debido a que generalmente demoran mucho más que los otros métodos. Como se puede observar a simple vista, Armadillo tuvo mejor eficiencia en la resolución de la matriz de 289x289 para los métodos LU, Cholesky y QR, no así para los métodos Mínimos Cuadrados, Givens ni Seidel.
En la figura 8 que tiene los gráficos de tiempo para la matriz de 1089x1089, se puede observar un comportamiento muy distinto al anterior, esta vez Matlab tiene mejor eficiencia en casi todos los casos, excepto Cholesky. En los casos de Mínimos Cuadrados, Seidel y LU, Matlab es mucho más eficiente.
Finalmente, en la figura 9 con los tiempos para la matriz de 4225x4225, se tiene un comportamiento similar al anterior, pero esta vez los métodos implementados en Armadillo demoran más en todos los casos respecto a Matlab.
Cabe destacar que se utilizaron matrices dispersas para la resolución de Givens en Armadillo, debido a que su tiempo de resolución sin el uso de matrices dispersas era del triple aproximadamente (200 segundos para la matriz

de 289x289 vs los 70 segundos obtenidos con matrices disperas para la matriz de 289x289).

3. Anexo

Cuadro 1: Comparativa de errores, matlab vs armadillo

Matriz	Método	Matlab	Armadillo
289x289	LU	$1{,}020e^{-15}$	$1{,}529e^{-15}$
	Cholesky	$1{,}588e^{-15}$	$6{,}021e^{-16}$
	QR	$2{,}982e^{-15}$	$5{,}149e^{-16}$
	Givens	$7{,}352e^{-16}$	$9{,}925e^{-16}$
	Mínimos Cuadrados	$1{,}508e^{-14}$	$4{,}785e^{-15}$
	Gauss Seidel	$5{,}748e^{-19}$	$5{,}748e^{-19}$
1089x1089	LU	$2{,}648e^{-15}$	$3{,}159e^{-15}$
	Cholesky	$8{,}390e^{-15}$	$2{,}116e^{-15}$
	QR	$1{,}159e^{-14}$	$1{,}905e^{-15}$
	Givens	$2{,}517e^{-15}$	$2{,}175e^{-15}$
	Mínimos Cuadrados	$1{,}508e^{-14}$	$6{,}054e^{-14}$
	Gauss Seidel	$4{,}554e^{-19}$	$9{,}108e^{-19}$
4225x4225	LU	$1{,}286e^{-14}$	$1{,}219e^{-14}$
	Cholesky	$5{,}417e^{-14}$	$6{,}838e^{-15}$
	QR	$8{,}376e^{-14}$	$6{,}923e^{-15}$
	Givens	$1{,}287e^{-14}$	$1{,}074e^{-14}$
	Mínimos Cuadrados	$7{,}686e^{-12}$	$8{,}348e^{-13}$
	Gauss Seidel	$2{,}371e^{-15}$	$2{,}322e^{-15}$

Cuadro 2: Número de Iteraciones y Error, Métodos Iterativos

Métodos	289x289	1089x1089	4225x4225
Gauss Seidel Armadillo (I)	94	348	1000
Gauss Seidel Matlab (I)	92	366	1000
Gauss Seidel Armadillo (E)	$5{,}748e^{-19}$	$9{,}108e^{-19}$	$2{,}322e^{-15}$
Gauss Seidel Matlab (E)	$5{,}748e^{-19}$	$4{,}554e^{-19}$	$2{,}371e^{-15}$

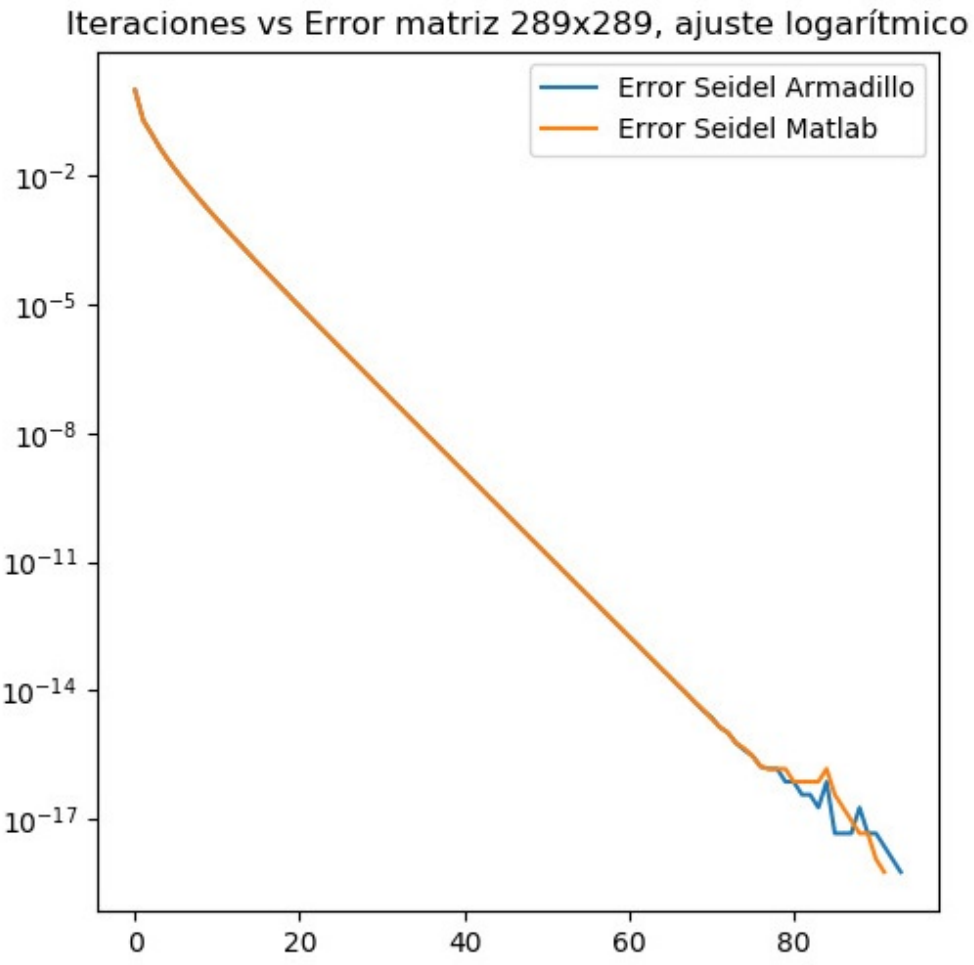


Figura 1: Comparativa de errores en Gauss Seidel, matlab vs armadillo

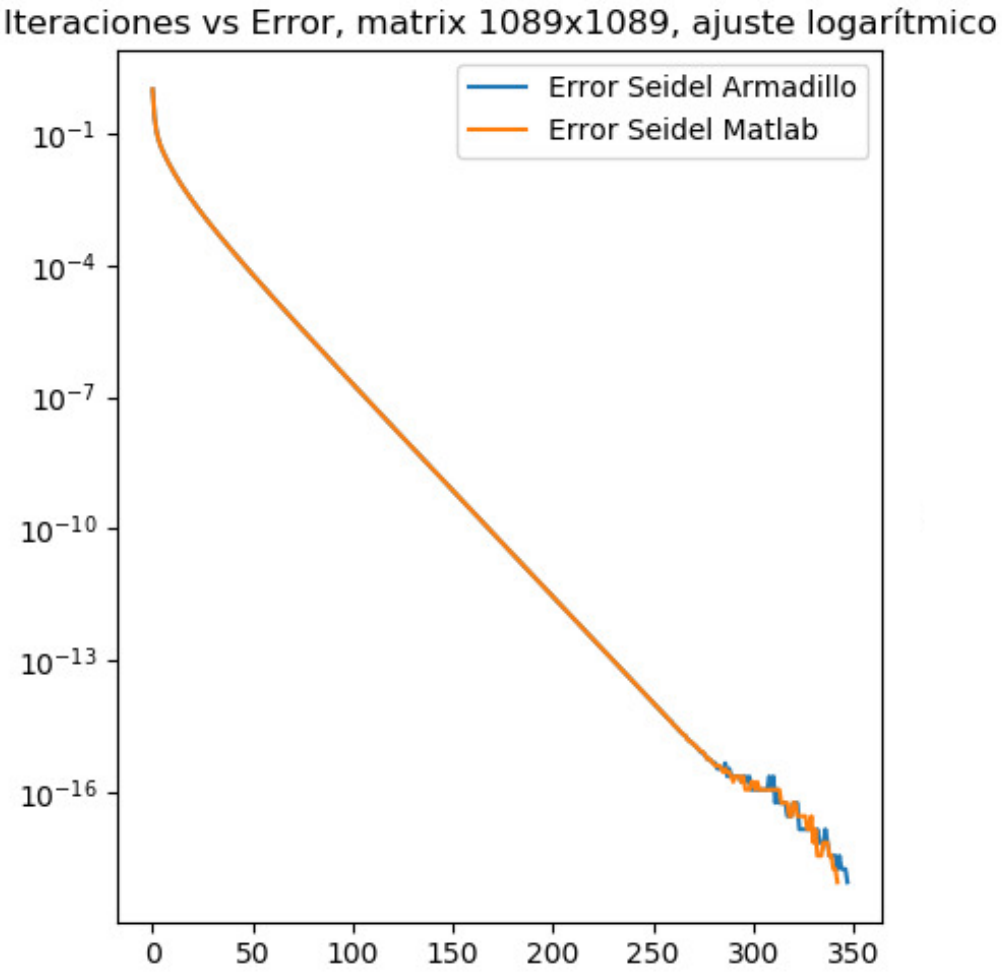


Figura 2: Comparativa de errores en Gauss Seidel, matlab vs armadillo

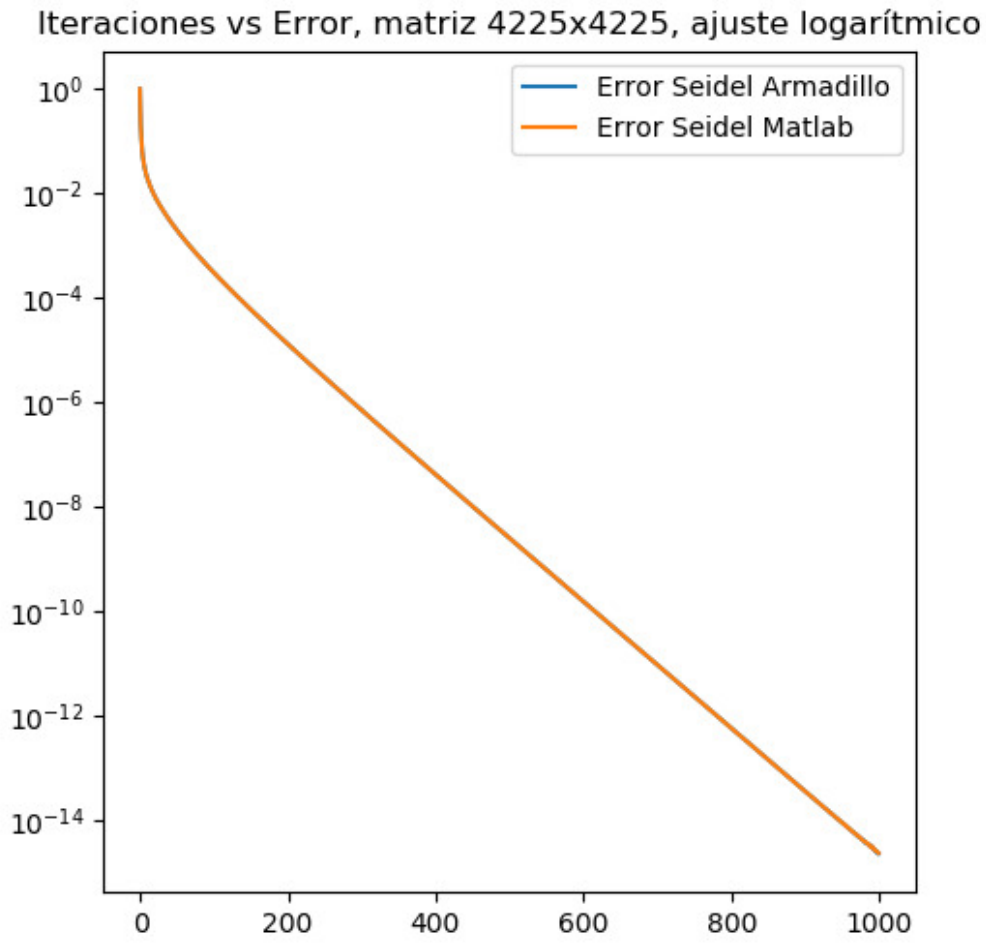


Figura 3: Comparativa de errores en Gauss Seidel, matlab vs armadillo

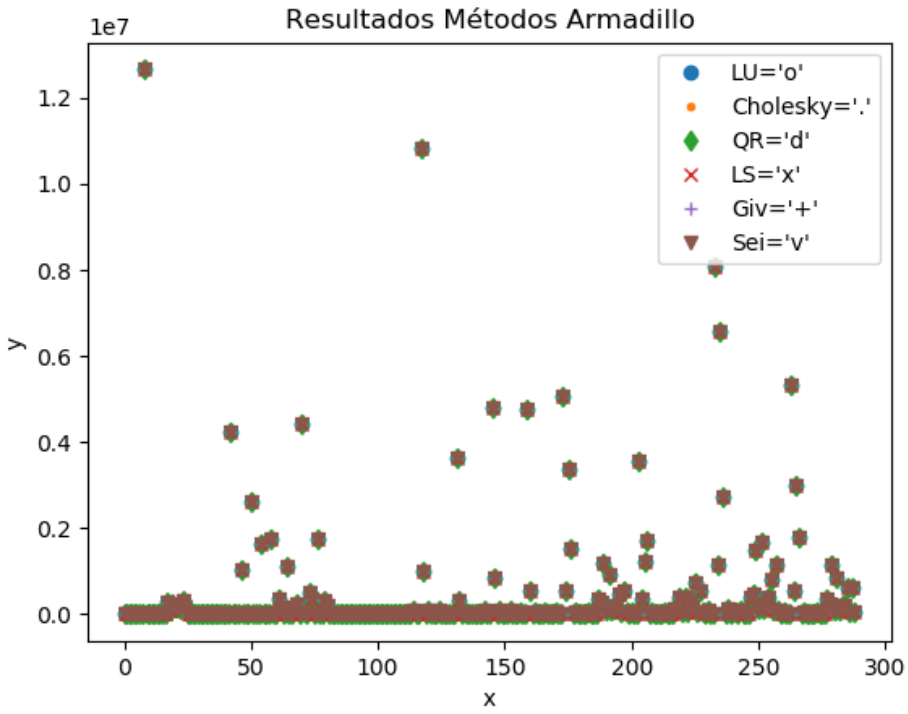


Figura 4: Resultados obtenidos en Armadillo, matriz 289x289

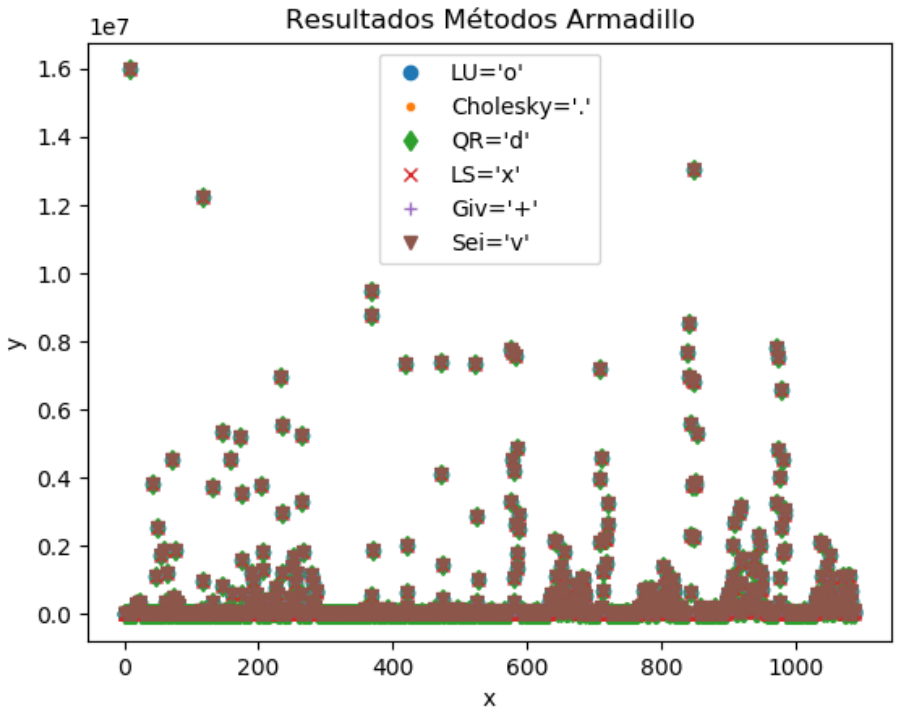


Figura 5: Resultados obtenidos en Armadillo, matriz 1089x1089

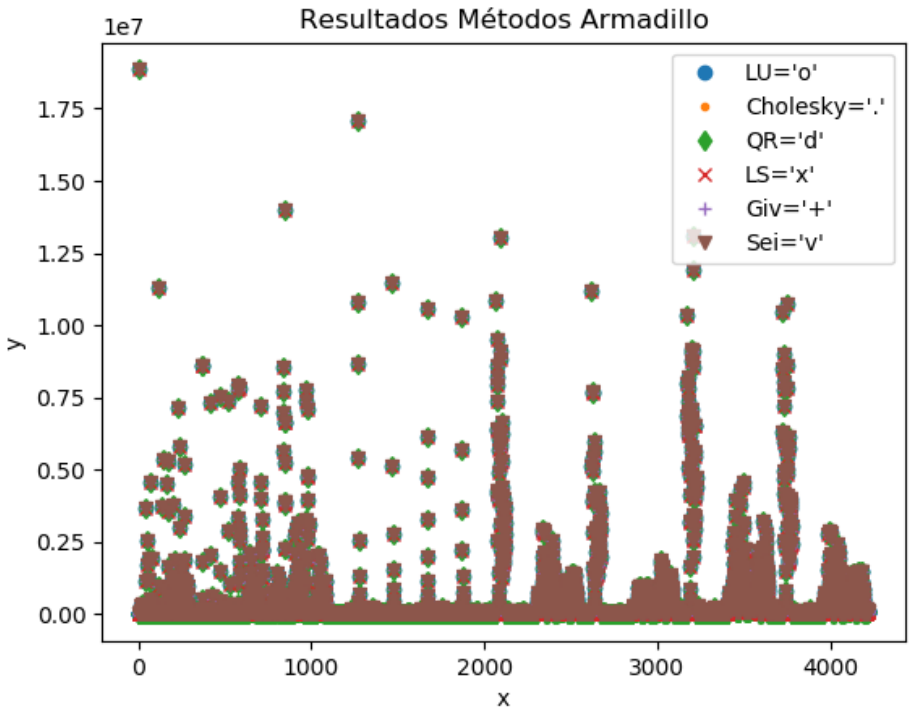
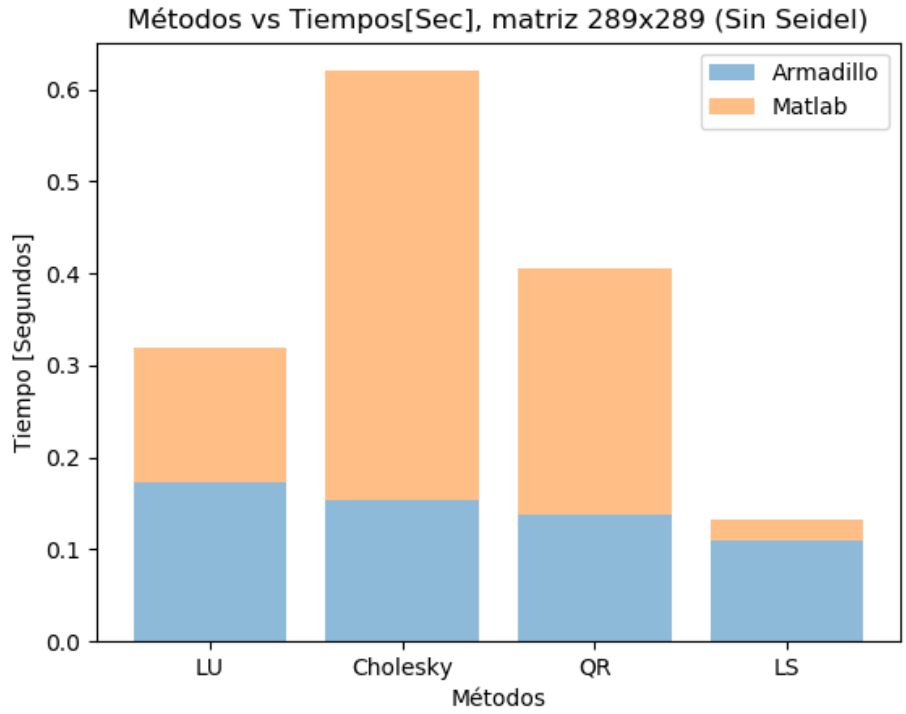


Figura 6: Resultados obtenidos en Armadillo, matriz 4225x4225



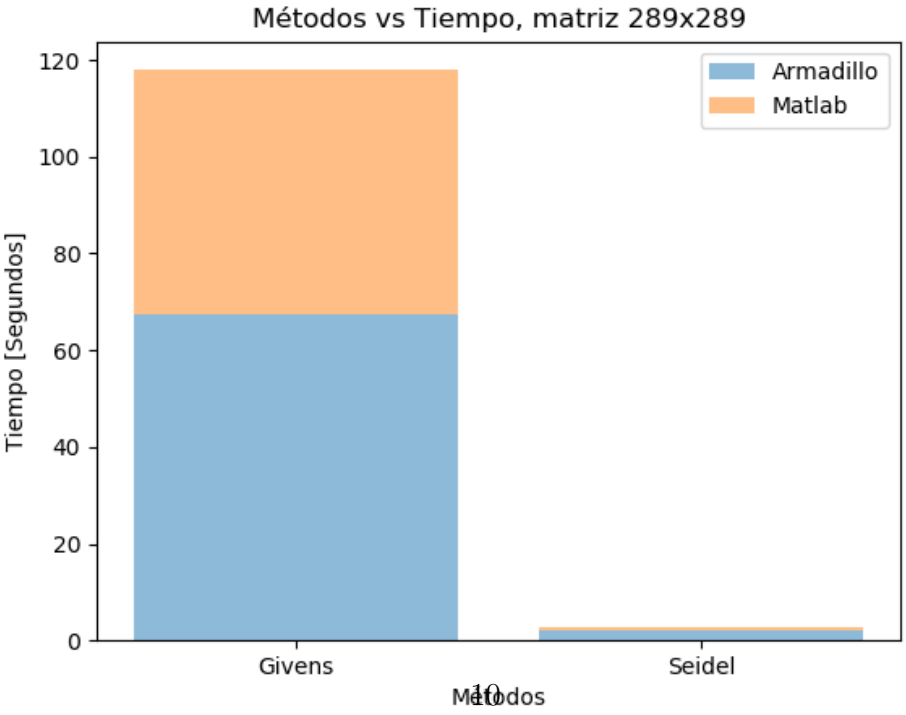
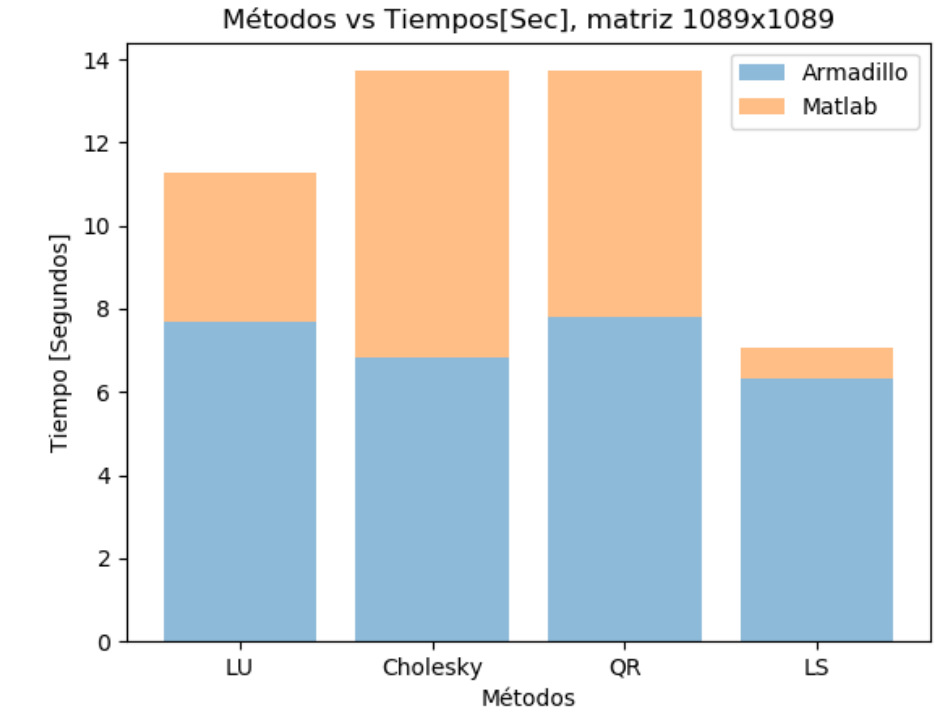


Figura 7: Tiempos Armadillo y Matlab, matriz 289x289



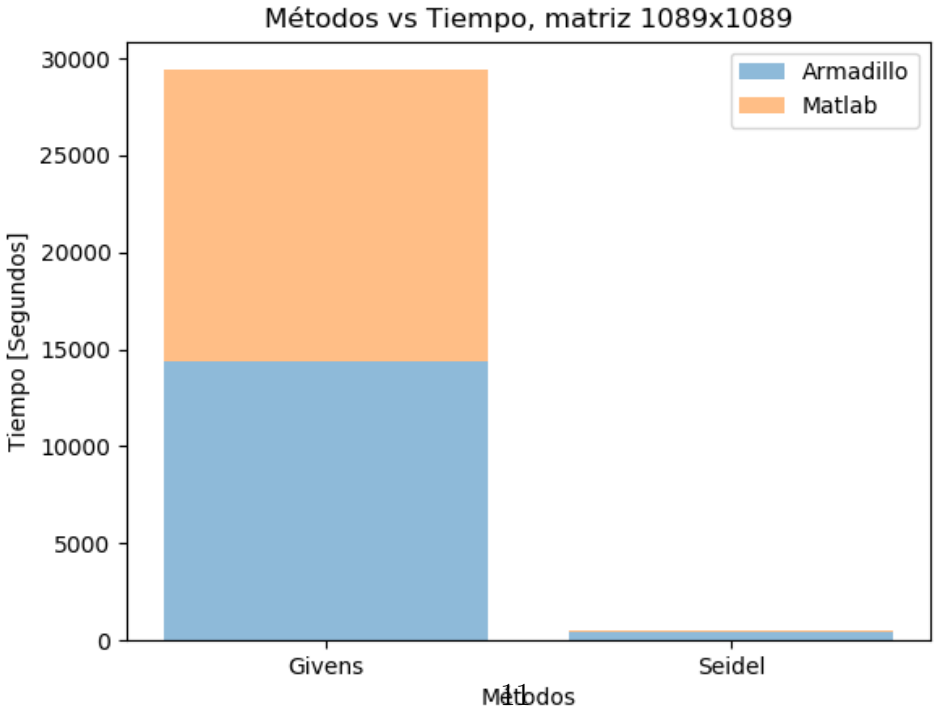
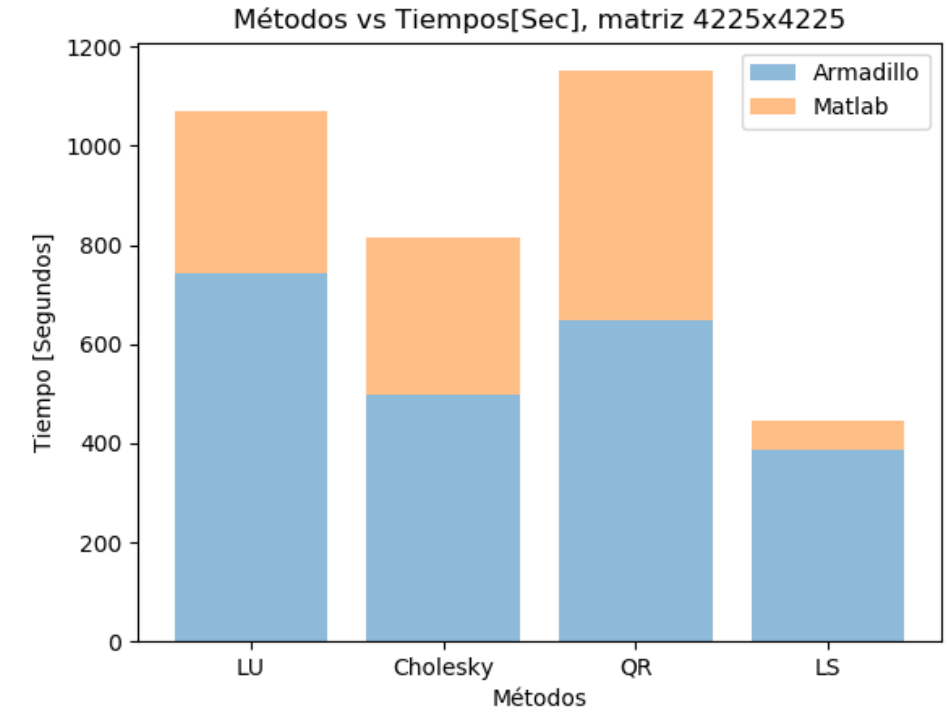


Figura 8: Tiempos Armadillo y Matlab, matriz 1089x1089



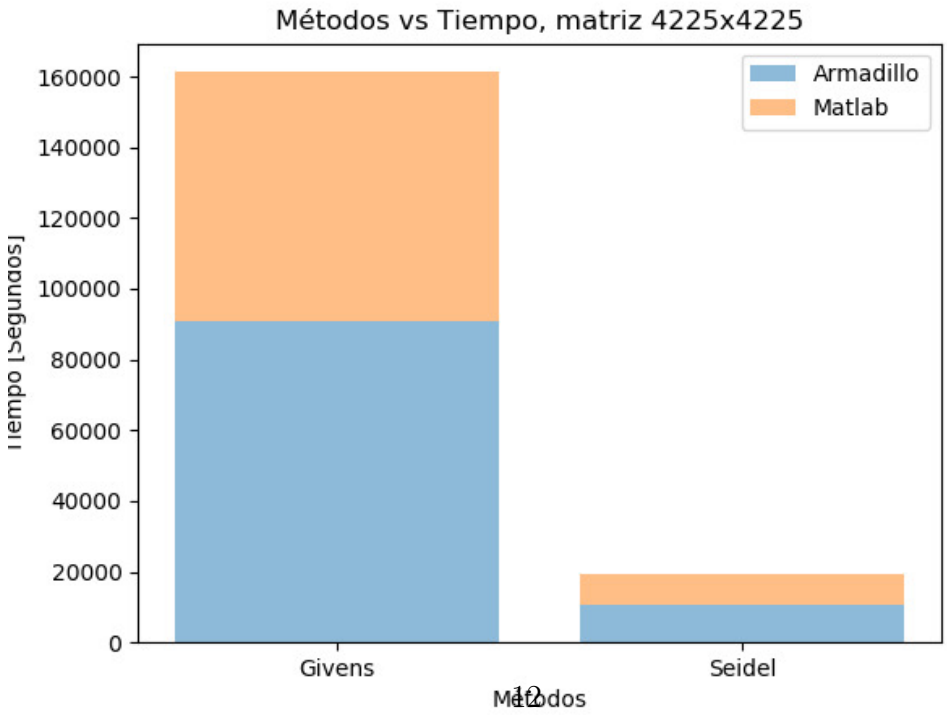


Figura 9: Tiempos Armadillo y Matlab, matriz 4225x4225