Bench report : 25-Feb-2013 à 9h42m22s

Monday 25th February, 2013

Contents

1 Introduction

Soient $\Omega \subset \mathbb{R}^2$ et $\boldsymbol{u}$, $\boldsymbol{v}$ deux champs de vecteurs (Vector Fields) On suppose que $\boldsymbol{u} = (u_1, u_2) \in \left(H^1(\Omega)\right)^2 = H^1(\Omega) \times H^1(\Omega)$ et $\boldsymbol{v} = (v_1, v_2) \in \left(H^1(\Omega)\right)^2 = H^1(\Omega) \times H^1(\Omega)$.

1.1 Matrice de masse en champs de vecteurs

On définie l'application bilinéaire $\mathcal{L}$ par $\mathcal{L}(\boldsymbol{u}, \boldsymbol{v}) = \langle \boldsymbol{u}, \boldsymbol{v} \rangle$.

La matrice de masse en champs de vecteurs est la matrice obtenue lors de la discrétisation par la méthode des éléments finis P_1-Lagrange de l'intégrale

$$\int_\Omega \mathcal{L}(\boldsymbol{u}, \boldsymbol{v})(\mathrm{q}) d\,\mathrm{q}\,.$$

1.2 Matrice de rigidité élastique en déformation plane

On note

$$\underline{\boldsymbol{\sigma}} = \begin{pmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{12} \end{pmatrix} \quad \text{et} \quad \underline{\boldsymbol{\epsilon}} = \begin{pmatrix} \epsilon_{11} \\ \epsilon_{22} \\ 2\epsilon_{12} \end{pmatrix}$$

respectivement le tenseur des contraintes vectorisé et le tenseur des déformations vectorisé en déformation plane.

On note $\mathcal{D}$ l'opérateur différentiel qui relie le déplacement aux déformations :

$$\underline{\boldsymbol{\epsilon}}(\boldsymbol{u}) = \mathcal{D}(\boldsymbol{u}) = \frac{1}{2}\left(\nabla(\boldsymbol{u}) + \nabla^t(\boldsymbol{u})\right) \tag{1}$$

Ceci donne en notation vectorielle, après réduction au plan,

$$\mathcal{D} = \begin{pmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{pmatrix}. \tag{2}$$

De plus, la loi de comportement s'écrit

$$\underline{\boldsymbol{\sigma}} = \mathbb{C}\underline{\boldsymbol{\epsilon}} = \begin{pmatrix} \lambda + 2\mu & \lambda & 0 \\ \lambda & \lambda + 2\mu & 0 \\ 0 & 0 & \mu \end{pmatrix} \underline{\boldsymbol{\epsilon}} \tag{3}$$

On définie l'application bilinéaire $\mathcal{L}$ par $\mathcal{L}(\boldsymbol{u}, \boldsymbol{v}) = \underline{\boldsymbol{\epsilon}}^t(\boldsymbol{u})\underline{\boldsymbol{\sigma}}(\boldsymbol{v}) = \underline{\boldsymbol{\epsilon}}^t(\boldsymbol{u})\mathbb{C}\underline{\boldsymbol{\epsilon}}(\boldsymbol{u})$.

La matrice de rigidité élastique est la matrice obtenue lors de la discrétisation par la méthode des éléments finis P_1-Lagrange de l'intégrale

$$\int_\Omega \mathcal{L}(\boldsymbol{u}, \boldsymbol{v})(\mathrm{q}) d\,\mathrm{q}\,.$$

2 Mesh : carre4-1

2.1 Function : MassVFAssembling

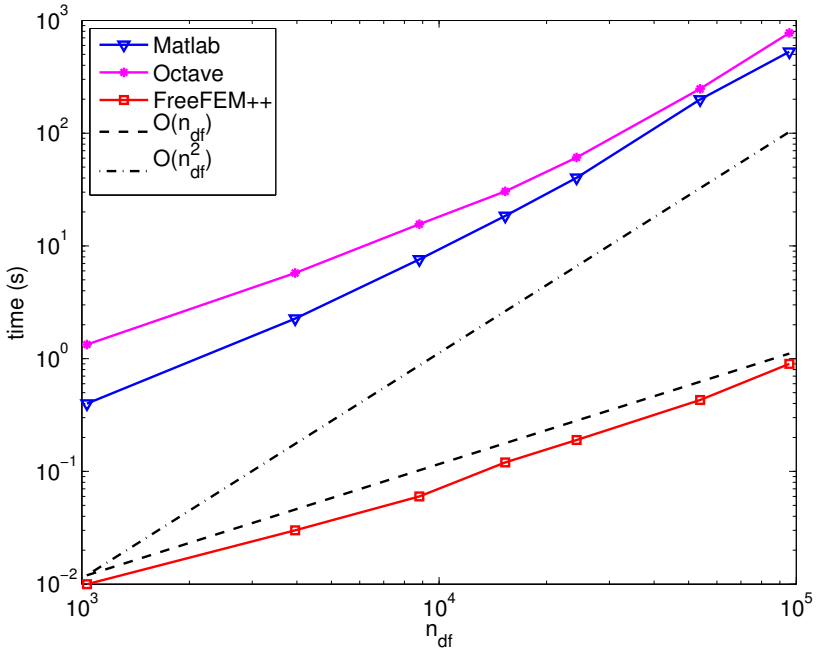

Figure 1: Comparison of Matlab/Octave function `MassVFAssemblingbase` and FreeFEM++

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
517	1034	0.400 (s) x 1.00	1.334 (s) x 0.30	0.010 (s) x 39.97
1978	3956	2.265 (s) x 1.00	5.753 (s) x 0.39	0.030 (s) x 75.51
4406	8812	7.583 (s) x 1.00	15.594 (s) x 0.49	0.060 (s) x 126.38
7662	15324	18.438 (s) x 1.00	30.447 (s) x 0.61	0.120 (s) x 153.65
12139	24278	40.228 (s) x 1.00	60.849 (s) x 0.66	0.190 (s) x 211.73
26910	53820	199.176 (s) x 1.00	247.272 (s) x 0.81	0.430 (s) x 463.20
47834	95668	527.715 (s) x 1.00	775.600 (s) x 0.68	0.900 (s) x 586.35

Table 1: Computational cost of the `MassVF` matrix assembly versus n_q/n_{df}, with the `base` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is base Matlab version.

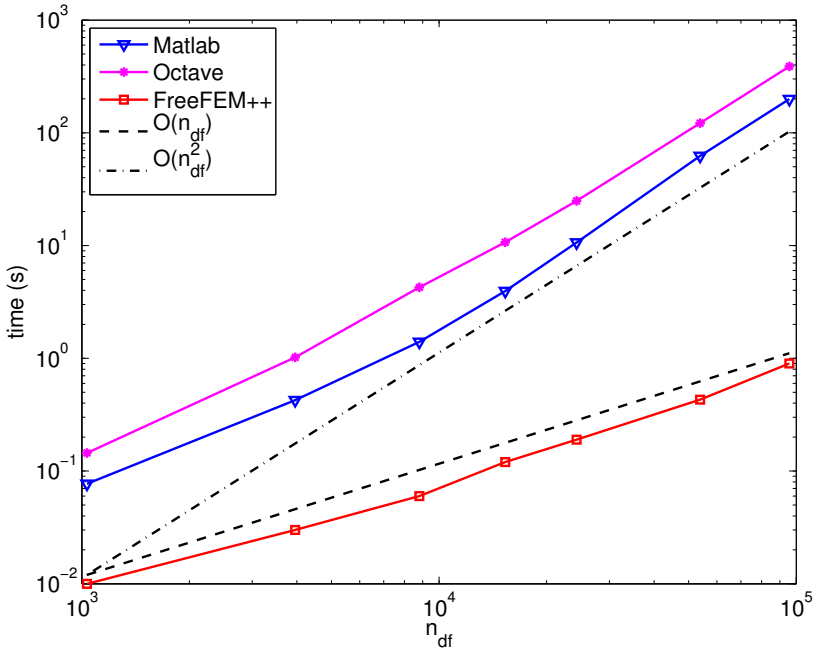

Figure 2: Comparison of Matlab/Octave function `MassVFAssemblingOptV0` and FreeFEM++

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
517	1034	0.077 (s) x 1.00	0.144 (s) x 0.53	0.010 (s) x 7.71
1978	3956	0.426 (s) x 1.00	1.018 (s) x 0.42	0.030 (s) x 14.19
4406	8812	1.398 (s) x 1.00	4.264 (s) x 0.33	0.060 (s) x 23.30
7662	15324	3.949 (s) x 1.00	10.700 (s) x 0.37	0.120 (s) x 32.91
12139	24278	10.632 (s) x 1.00	24.877 (s) x 0.43	0.190 (s) x 55.96
26910	53820	62.367 (s) x 1.00	121.741 (s) x 0.51	0.430 (s) x 145.04
47834	95668	199.206 (s) x 1.00	388.305 (s) x 0.51	0.900 (s) x 221.34

Table 2: Computational cost of the `MassVF` matrix assembly versus n_q/n_{df}, with the `OptV0` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is OptV0 Matlab version.

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
517	1034	0.023 (s) x 1.00	0.113 (s) x 0.20	0.010 (s) x 2.25
1978	3956	0.092 (s) x 1.00	0.445 (s) x 0.21	0.030 (s) x 3.06
4406	8812	0.208 (s) x 1.00	1.505 (s) x 0.14	0.060 (s) x 3.47
7662	15324	0.367 (s) x 1.00	1.774 (s) x 0.21	0.120 (s) x 3.06
12139	24278	0.591 (s) x 1.00	3.825 (s) x 0.15	0.190 (s) x 3.11
26910	53820	1.321 (s) x 1.00	6.291 (s) x 0.21	0.430 (s) x 3.07
47834	95668	2.396 (s) x 1.00	12.247 (s) x 0.20	0.900 (s) x 2.66

Table 3: Computational cost of the `MassVF` matrix assembly versus n_q/n_{df}, with the `OptV1` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is OptV1 Matlab version.

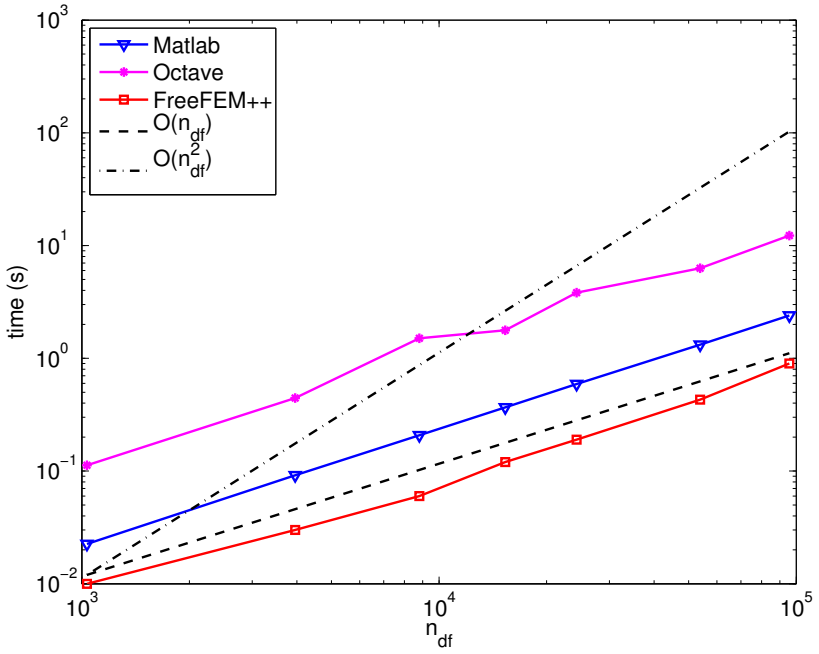

Figure 3: Comparison of Matlab/Octave function `MassVFAssemblingOptV1` and FreeFEM++

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
517	1034	0.010 (s) x 1.00	0.003 (s) x 3.33	0.010 (s) x 1.01
1978	3956	0.023 (s) x 1.00	0.009 (s) x 2.59	0.030 (s) x 0.76
4406	8812	0.048 (s) x 1.00	0.025 (s) x 1.95	0.060 (s) x 0.80
7662	15324	0.086 (s) x 1.00	0.035 (s) x 2.46	0.120 (s) x 0.71
12139	24278	0.143 (s) x 1.00	0.056 (s) x 2.55	0.190 (s) x 0.75
26910	53820	0.331 (s) x 1.00	0.159 (s) x 2.08	0.430 (s) x 0.77
47834	95668	0.611 (s) x 1.00	0.290 (s) x 2.10	0.900 (s) x 0.68

Table 4: Computational cost of the `MassVF` matrix assembly versus n_q/n_{df}, with the `OptV2` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is OptV2 Matlab version.

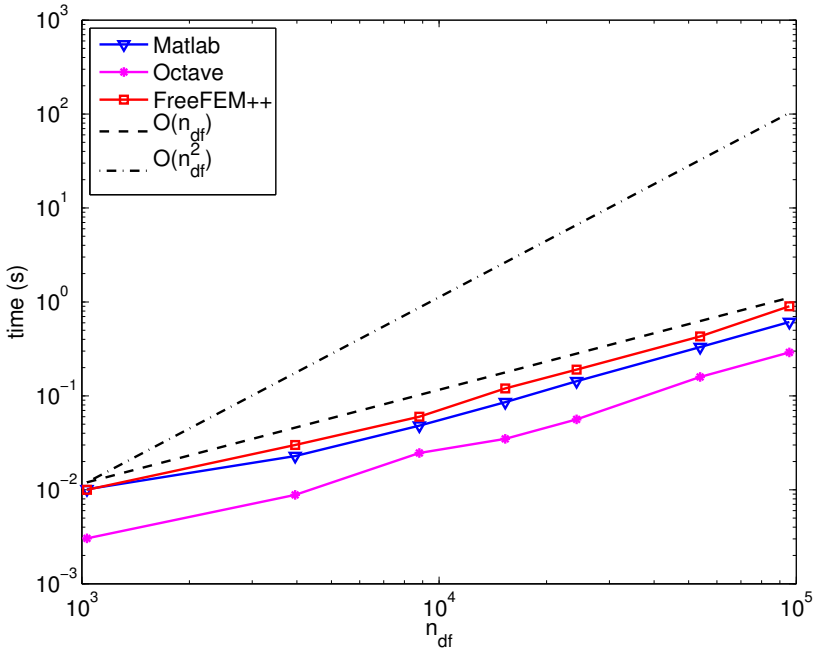

Figure 4: Comparison of Matlab/Octave function `MassVFAssemblingOptV2` and FreeFEM++

2.2 Function : StiffElasAssembling

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
517	1034	0.464 (s) x 1.00	1.571 (s) x 0.30	0.040 (s) x 11.61
1978	3956	3.322 (s) x 1.00	7.818 (s) x 0.42	0.170 (s) x 19.54
4406	8812	13.907 (s) x 1.00	26.072 (s) x 0.53	0.390 (s) x 35.66
7662	15324	36.122 (s) x 1.00	65.811 (s) x 0.55	0.680 (s) x 53.12
12139	24278	77.035 (s) x 1.00	166.429 (s) x 0.46	1.070 (s) x 72.00
26910	53820	490.139 (s) x 1.00	862.389 (s) x 0.57	2.380 (s) x 205.94
47834	95668	1087.304 (s) x 1.00	2919.172 (s) x 0.37	4.500 (s) x 241.62

Table 5: Computational cost of the `StiffElas` matrix assembly versus n_q/n_{df}, with the `base` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is base Matlab version.

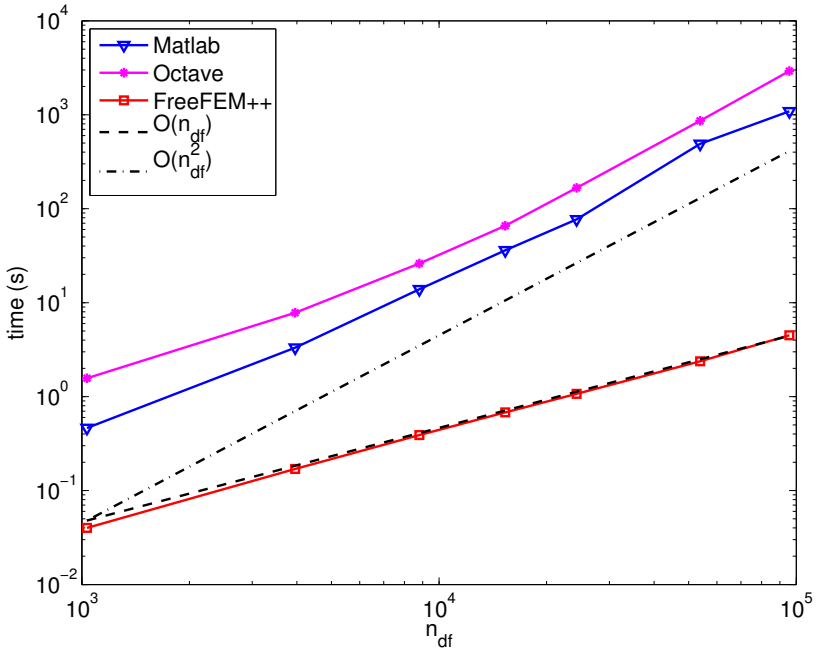

Figure 5: Comparison of Matlab/Octave function `StiffElasAssemblingbase` and FreeFEM++

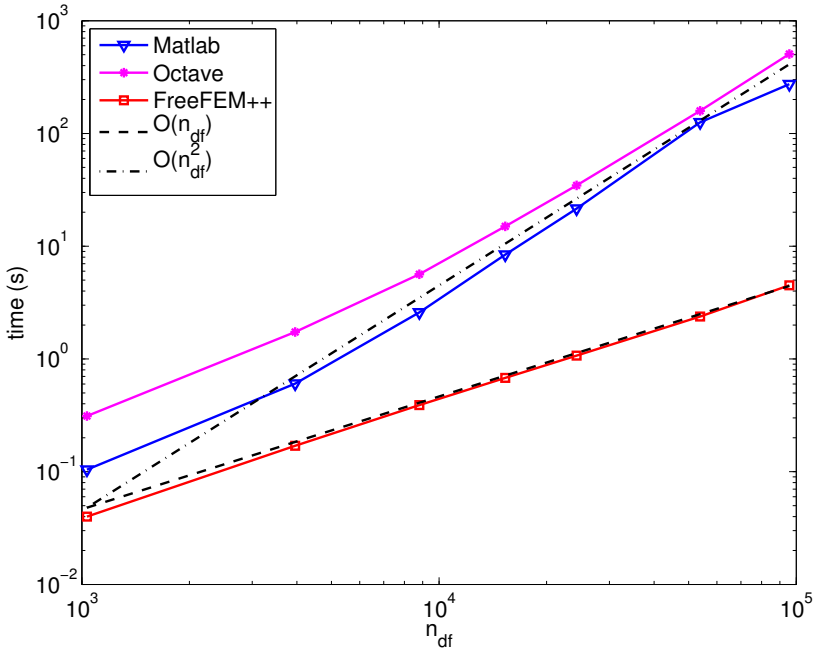

Figure 6: Comparison of Matlab/Octave function `StiffElasAssemblingOptV0` and FreeFEM++

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
517	1034	0.104 (s) x 1.00	0.312 (s) x 0.33	0.040 (s) x 2.61
1978	3956	0.604 (s) x 1.00	1.733 (s) x 0.35	0.170 (s) x 3.56
4406	8812	2.590 (s) x 1.00	5.637 (s) x 0.46	0.390 (s) x 6.64
7662	15324	8.409 (s) x 1.00	15.013 (s) x 0.56	0.680 (s) x 12.37
12139	24278	21.553 (s) x 1.00	34.646 (s) x 0.62	1.070 (s) x 20.14
26910	53820	125.780 (s) x 1.00	158.967 (s) x 0.79	2.380 (s) x 52.85
47834	95668	273.629 (s) x 1.00	506.596 (s) x 0.54	4.500 (s) x 60.81

Table 6: Computational cost of the `StiffElas` matrix assembly versus n_q/n_{df}, with the `OptV0` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is OptV0 Matlab version.

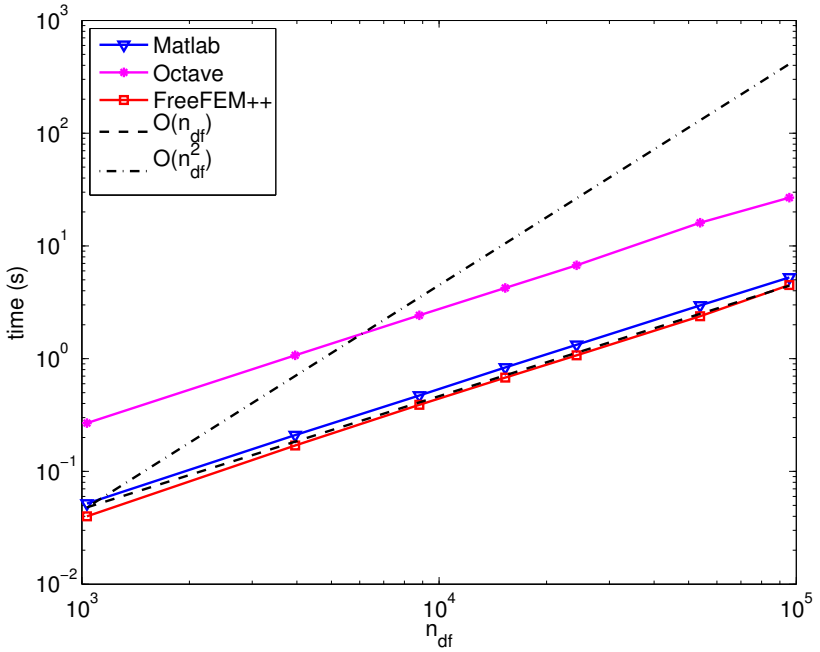

Figure 7: Comparison of Matlab/Octave function `StiffElasAssemblingOptV1` and FreeFEM++

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
517	1034	0.052 (s) x 1.00	0.268 (s) x 0.19	0.040 (s) x 1.30
1978	3956	0.210 (s) x 1.00	1.071 (s) x 0.20	0.170 (s) x 1.23
4406	8812	0.471 (s) x 1.00	2.426 (s) x 0.19	0.390 (s) x 1.21
7662	15324	0.836 (s) x 1.00	4.234 (s) x 0.20	0.680 (s) x 1.23
12139	24278	1.329 (s) x 1.00	6.740 (s) x 0.20	1.070 (s) x 1.24
26910	53820	2.963 (s) x 1.00	16.047 (s) x 0.18	2.380 (s) x 1.25
47834	95668	5.263 (s) x 1.00	26.789 (s) x 0.20	4.500 (s) x 1.17

Table 7: Computational cost of the `StiffElas` matrix assembly versus n_q/n_{df}, with the `OptV1` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is OptV1 Matlab version.

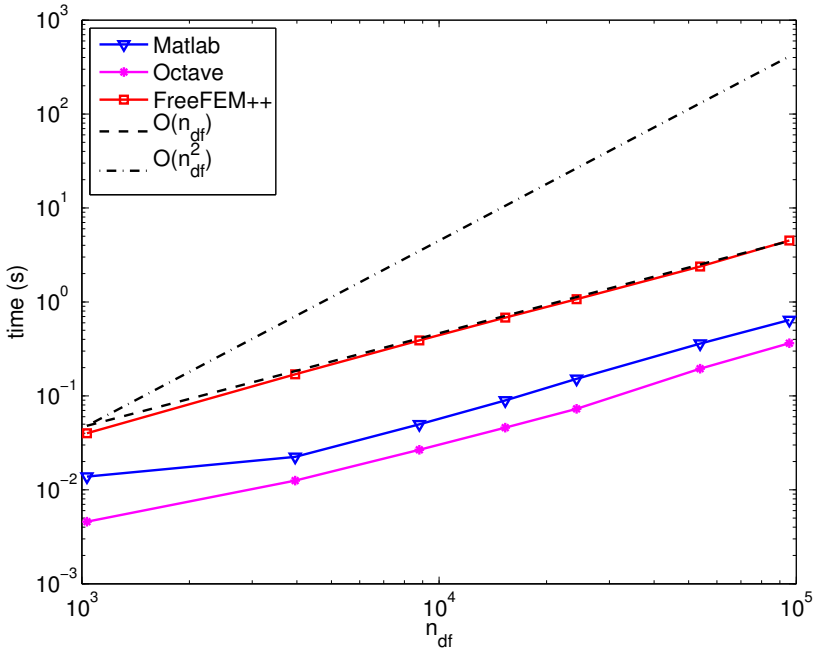

Figure 8: Comparison of Matlab/Octave function `StiffElasAssemblingOptV2` and FreeFEM++

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
517	1034	0.014 (s) x 1.00	0.005 (s) x 3.01	0.040 (s) x 0.35
1978	3956	0.022 (s) x 1.00	0.013 (s) x 1.79	0.170 (s) x 0.13
4406	8812	0.050 (s) x 1.00	0.027 (s) x 1.86	0.390 (s) x 0.13
7662	15324	0.090 (s) x 1.00	0.046 (s) x 1.95	0.680 (s) x 0.13
12139	24278	0.152 (s) x 1.00	0.073 (s) x 2.09	1.070 (s) x 0.14
26910	53820	0.360 (s) x 1.00	0.195 (s) x 1.85	2.380 (s) x 0.15
47834	95668	0.643 (s) x 1.00	0.363 (s) x 1.77	4.500 (s) x 0.14

Table 8: Computational cost of the `StiffElas` matrix assembly versus n_q/n_{df}, with the `OptV2` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is OptV2 Matlab version.

3 Mesh : disque4-1

3.1 Function : MassVFAssembling

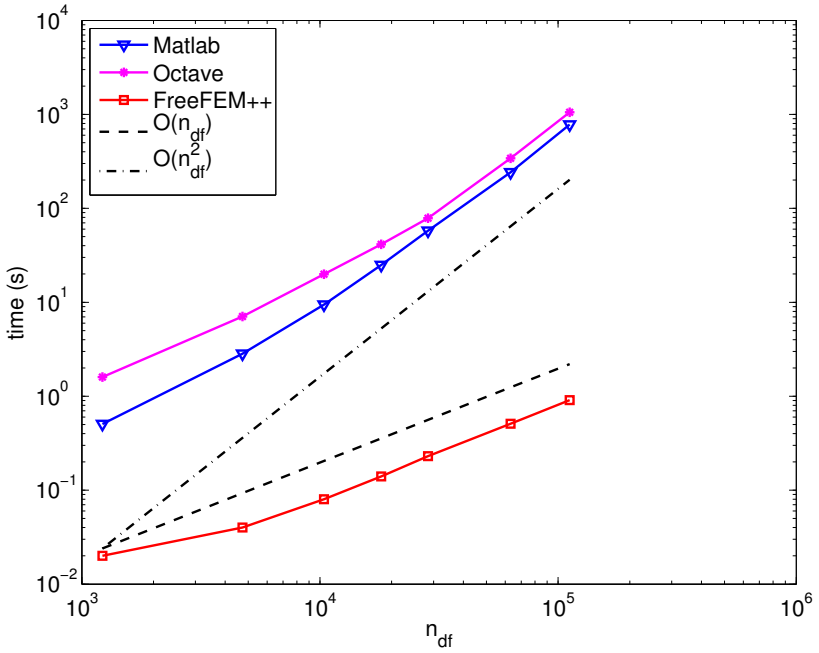

Figure 9: Comparison of Matlab/Octave function `MassVFAssemblingbase` and FreeFEM++

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
610	1220	0.506 (s) x 1.00	1.601 (s) x 0.32	0.020 (s) x 25.28
2365	4730	2.833 (s) x 1.00	7.069 (s) x 0.40	0.040 (s) x 70.83
5199	10398	9.455 (s) x 1.00	19.882 (s) x 0.48	0.080 (s) x 118.19
9034	18068	24.936 (s) x 1.00	41.390 (s) x 0.60	0.140 (s) x 178.12
14222	28444	57.809 (s) x 1.00	78.650 (s) x 0.74	0.230 (s) x 251.35
31575	63150	241.760 (s) x 1.00	340.554 (s) x 0.71	0.510 (s) x 474.04
55919	111838	780.439 (s) x 1.00	1054.403 (s) x 0.74	0.910 (s) x 857.62

Table 9: Computational cost of the `MassVF` matrix assembly versus n_q/n_{df}, with the `base` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is base Matlab version.

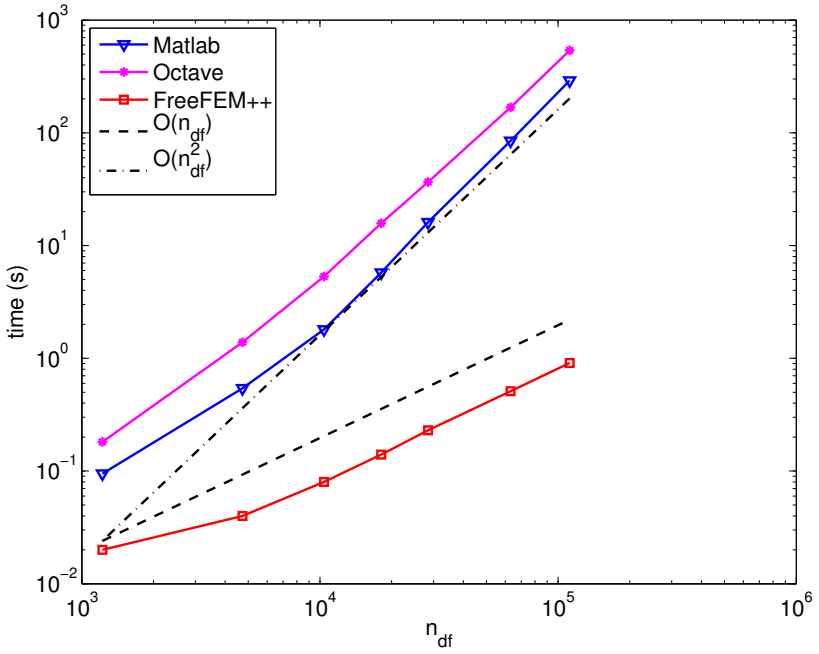

Figure 10: Comparison of Matlab/Octave function `MassVFAssemblingOptV0` and FreeFEM++

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
610	1220	0.095 (s) x 1.00	0.181 (s) x 0.52	0.020 (s) x 4.74
2365	4730	0.541 (s) x 1.00	1.388 (s) x 0.39	0.040 (s) x 13.53
5199	10398	1.799 (s) x 1.00	5.316 (s) x 0.34	0.080 (s) x 22.49
9034	18068	5.757 (s) x 1.00	15.768 (s) x 0.37	0.140 (s) x 41.12
14222	28444	16.084 (s) x 1.00	36.596 (s) x 0.44	0.230 (s) x 69.93
31575	63150	84.901 (s) x 1.00	168.191 (s) x 0.50	0.510 (s) x 166.47
55919	111838	290.565 (s) x 1.00	537.385 (s) x 0.54	0.910 (s) x 319.30

Table 10: Computational cost of the `MassVF` matrix assembly versus n_q/n_{df}, with the `OptV0` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is OptV0 Matlab version.

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
610	1220	0.027 (s) x 1.00	0.133 (s) x 0.21	0.020 (s) x 1.37
2365	4730	0.110 (s) x 1.00	0.535 (s) x 0.20	0.040 (s) x 2.74
5199	10398	0.248 (s) x 1.00	1.201 (s) x 0.21	0.080 (s) x 3.10
9034	18068	0.438 (s) x 1.00	2.091 (s) x 0.21	0.140 (s) x 3.13
14222	28444	0.691 (s) x 1.00	3.306 (s) x 0.21	0.230 (s) x 3.00
31575	63150	1.581 (s) x 1.00	7.362 (s) x 0.21	0.510 (s) x 3.10
55919	111838	2.813 (s) x 1.00	13.195 (s) x 0.21	0.910 (s) x 3.09

Table 11: Computational cost of the `MassVF` matrix assembly versus n_q/n_{df}, with the `OptV1` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is OptV1 Matlab version.

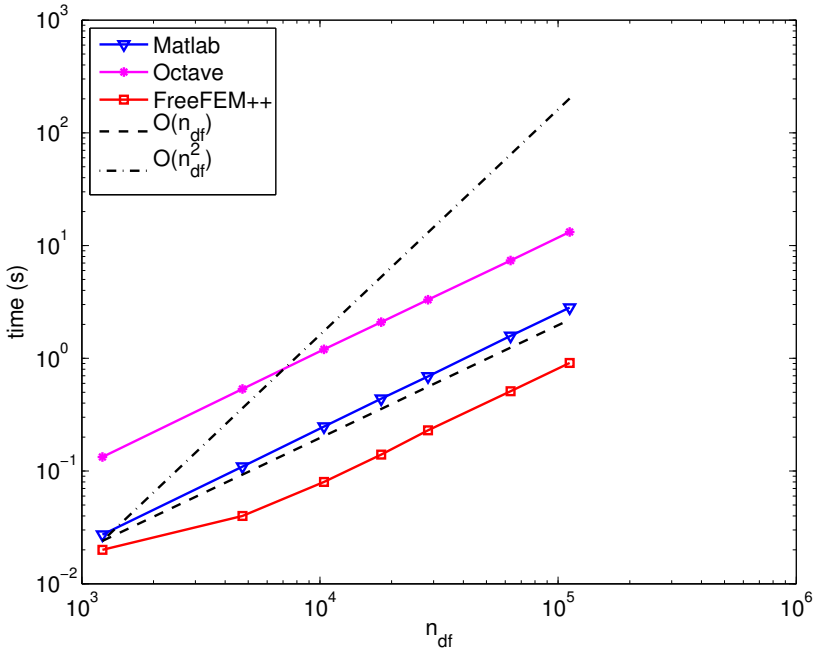

Figure 11: Comparison of Matlab/Octave function `MassVFAssemblingOptV1` and FreeFEM++

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
610	1220	0.006 (s) x 1.00	0.003 (s) x 1.93	0.020 (s) x 0.32
2365	4730	0.027 (s) x 1.00	0.011 (s) x 2.53	0.040 (s) x 0.67
5199	10398	0.056 (s) x 1.00	0.024 (s) x 2.40	0.080 (s) x 0.71
9034	18068	0.106 (s) x 1.00	0.041 (s) x 2.56	0.140 (s) x 0.76
14222	28444	0.168 (s) x 1.00	0.067 (s) x 2.50	0.230 (s) x 0.73
31575	63150	0.395 (s) x 1.00	0.192 (s) x 2.06	0.510 (s) x 0.77
55919	111838	0.724 (s) x 1.00	0.333 (s) x 2.17	0.910 (s) x 0.80

Table 12: Computational cost of the `MassVF` matrix assembly versus n_q/n_{df}, with the `OptV2` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is OptV2 Matlab version.

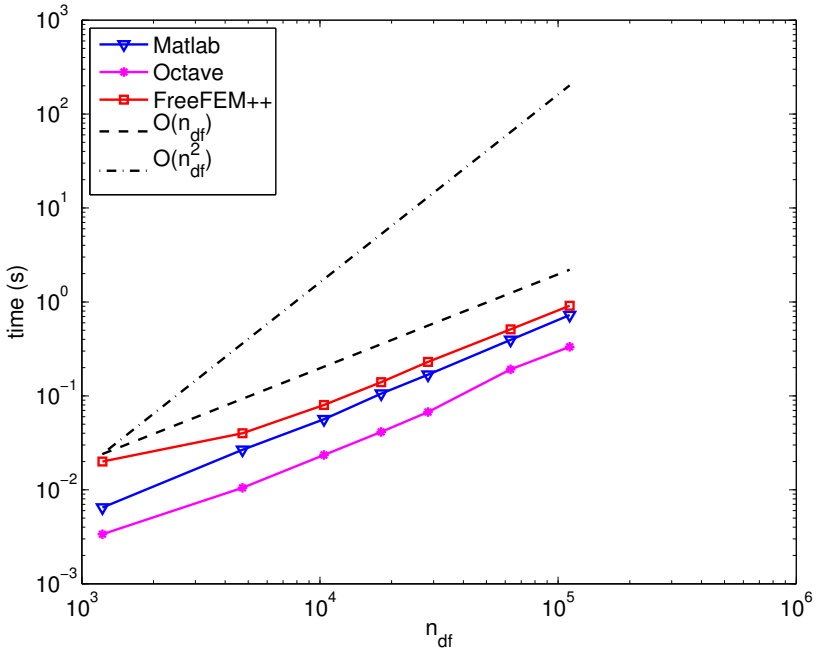

Figure 12: Comparison of Matlab/Octave function `MassVFAssemblingOptV2` and FreeFEM++

3.2 Function : StiffElasAssembling

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
610	1220	0.609 (s) x 1.00	1.901 (s) x 0.32	0.050 (s) x 12.18
2365	4730	4.000 (s) x 1.00	9.920 (s) x 0.40	0.210 (s) x 19.05
5199	10398	16.253 (s) x 1.00	33.942 (s) x 0.48	0.450 (s) x 36.12
9034	18068	45.750 (s) x 1.00	90.202 (s) x 0.51	0.800 (s) x 57.19
14222	28444	108.713 (s) x 1.00	225.183 (s) x 0.48	1.250 (s) x 86.97
31575	63150	502.183 (s) x 1.00	1192.036 (s) x 0.42	2.800 (s) x 179.35
55919	111838	1885.628 (s) x 1.00	4086.153 (s) x 0.46	4.970 (s) x 379.40

Table 13: Computational cost of the `StiffElas` matrix assembly versus n_q/n_{df}, with the `base` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is base Matlab version.

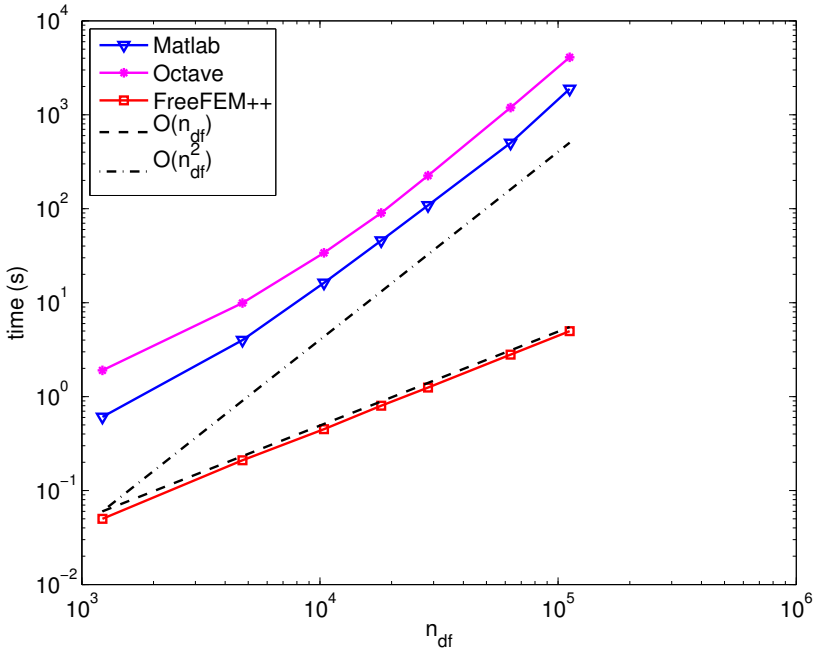

Figure 13: Comparison of Matlab/Octave function `StiffElasAssemblingbase` and FreeFEM++

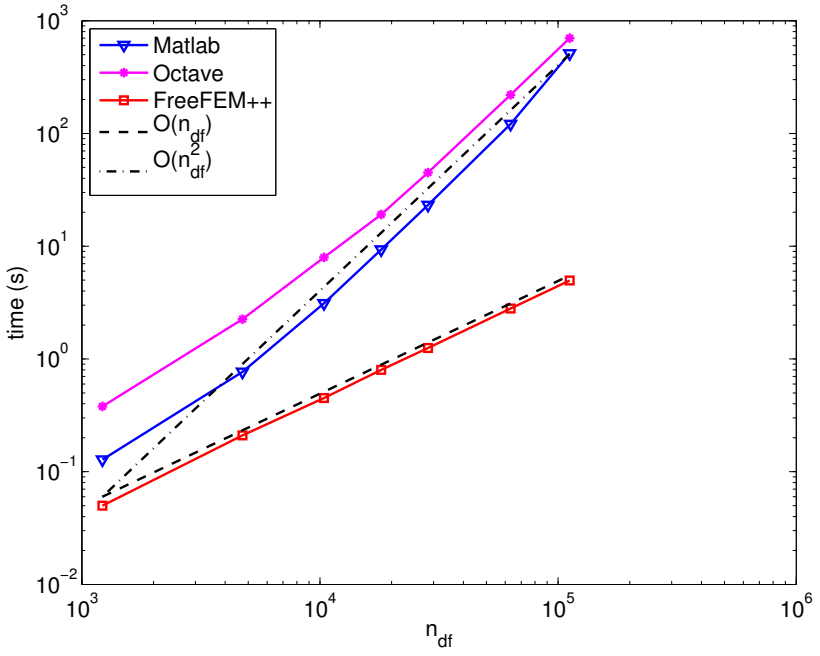

Figure 14: Comparison of Matlab/Octave function `StiffElasAssemblingOptV0` and FreeFEM++

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
610	1220	0.128 (s) x 1.00	0.380 (s) x 0.34	0.050 (s) x 2.55
2365	4730	0.768 (s) x 1.00	2.253 (s) x 0.34	0.210 (s) x 3.65
5199	10398	3.115 (s) x 1.00	7.961 (s) x 0.39	0.450 (s) x 6.92
9034	18068	9.350 (s) x 1.00	19.143 (s) x 0.49	0.800 (s) x 11.69
14222	28444	23.199 (s) x 1.00	45.026 (s) x 0.52	1.250 (s) x 18.56
31575	63150	121.342 (s) x 1.00	220.383 (s) x 0.55	2.800 (s) x 43.34
55919	111838	512.553 (s) x 1.00	700.898 (s) x 0.73	4.970 (s) x 103.13

Table 14: Computational cost of the `StiffElas` matrix assembly versus n_q/n_{df}, with the `OptV0` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is OptV0 Matlab version.

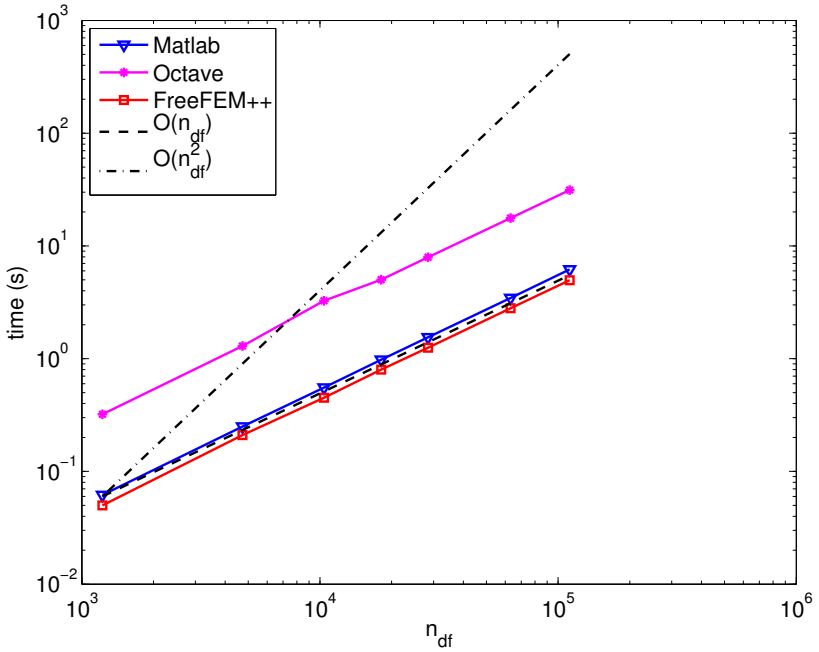

Figure 15: Comparison of Matlab/Octave function `StiffElasAssemblingOptV1` and FreeFEM++

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
610	1220	0.062 (s) x 1.00	0.321 (s) x 0.19	0.050 (s) x 1.24
2365	4730	0.250 (s) x 1.00	1.295 (s) x 0.19	0.210 (s) x 1.19
5199	10398	0.554 (s) x 1.00	3.260 (s) x 0.17	0.450 (s) x 1.23
9034	18068	0.980 (s) x 1.00	5.014 (s) x 0.20	0.800 (s) x 1.22
14222	28444	1.548 (s) x 1.00	7.920 (s) x 0.20	1.250 (s) x 1.24
31575	63150	3.456 (s) x 1.00	17.665 (s) x 0.20	2.800 (s) x 1.23
55919	111838	6.186 (s) x 1.00	31.291 (s) x 0.20	4.970 (s) x 1.24

Table 15: Computational cost of the `StiffElas` matrix assembly versus n_q/n_{df}, with the `OptV1` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is OptV1 Matlab version.

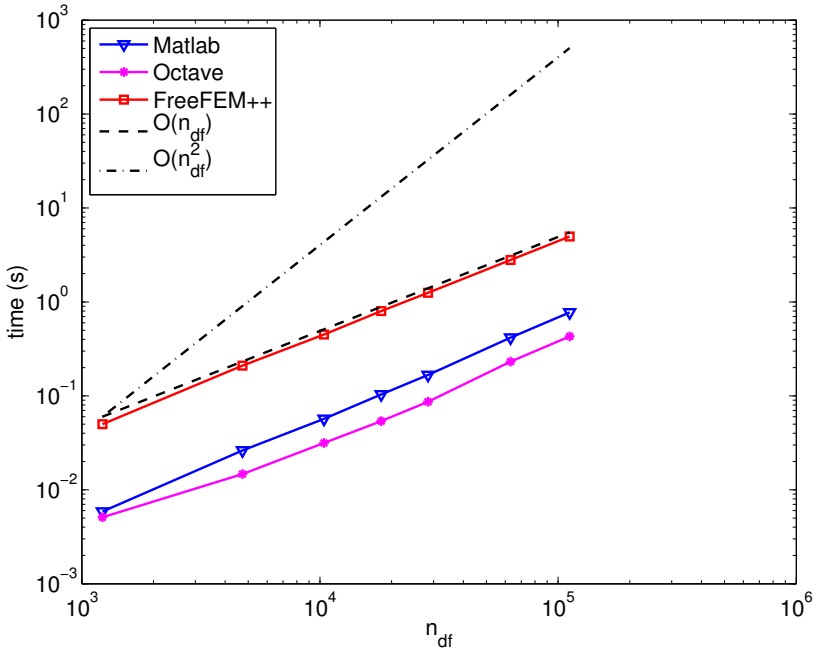

Figure 16: Comparison of Matlab/Octave function `StiffElasAssemblingOptV2` and FreeFEM++

n_q	n_{df}	Matlab (R2012b)	Octave (3.6.3)	FreeFEM++ (3.2)
610	1220	0.006 (s) x 1.00	0.005 (s) x 1.15	0.050 (s) x 0.12
2365	4730	0.026 (s) x 1.00	0.015 (s) x 1.78	0.210 (s) x 0.12
5199	10398	0.057 (s) x 1.00	0.032 (s) x 1.80	0.450 (s) x 0.13
9034	18068	0.103 (s) x 1.00	0.054 (s) x 1.92	0.800 (s) x 0.13
14222	28444	0.167 (s) x 1.00	0.087 (s) x 1.93	1.250 (s) x 0.13
31575	63150	0.417 (s) x 1.00	0.232 (s) x 1.80	2.800 (s) x 0.15
55919	111838	0.773 (s) x 1.00	0.429 (s) x 1.80	4.970 (s) x 0.16

Table 16: Computational cost of the `StiffElas` matrix assembly versus n_q/n_{df}, with the `OptV2` Matlab/Octave version ($3^{\text{rd}}/4^{\text{th}}$ columns) and with FreeFEM++ (5^{th} column) : time in seconds (top value) and speedup (bottom value). The speedup reference is OptV2 Matlab version.