Citation for published version:

Pegg, E & Gill, H 2016, 'An open source software tool to assign the material properties of bone for ABAQUS finite element simulations', *Journal of Biomechanics*, vol. 49, no. 13, pp. 3116-3121.
<https://doi.org/10.1016/j.jbiomech.2016.07.037>

DOI:

[10.1016/j.jbiomech.2016.07.037](https://doi.org/10.1016/j.jbiomech.2016.07.037)

Publication date:

2016

Document Version

Peer reviewed version

[Link to publication](#)

Publisher Rights

CC BY-NC-ND

University of Bath

Alternative formats

If you require this document in an alternative format, please contact:
openaccess@bath.ac.uk

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

1 **An open source software tool to assign the material properties of bone for**
2 **ABAQUS finite element simulations**

3 Elise C Pegg^{1*}, Harinderjit S Gill¹

4 ¹ Department of Mechanical Engineering, University of Bath, Bath, UK

5

6 * Corresponding Author:

7 Dr Elise C Pegg

8 Department of Mechanical Engineering

9 University of Bath, Bath

10 BA2 7AY

11 Tel: 01225 383375

12 Email: e.c.pegg@bath.ac.uk

13

14 **Keywords:** Material properties; Bone; Finite Element; Python.

15 **Word Count:** 1,989

16

17 **Abstract**

18 A new software tool to assign the material properties of bone to an ABAQUS finite element mesh was created and compared
19 with Bonemat, a similar tool originally designed to work with Ansys finite element models. Our software tool
20 (py_bonemat_abaqus) was written in Python, which is the chosen scripting language for ABAQUS. The purpose of this study
21 was to compare the software packages in terms of the material assignment calculation and processing speed. Three element
22 types were compared (linear hexahedral (C3D8), linear tetrahedral (C3D4) and quadratic tetrahedral elements (C3D10)),
23 both individually and as part of a mesh.

24 Comparisons were made using a CT scan of a hemi-pelvis as a test case. A small difference, of -0.05 kPa on average, was
25 found between Bonemat version 3.1 (the current version) and our python package. Errors were found in the previous
26 release of Bonemat (version 3.0 downloaded from www.biomedtown.org) during calculation of the quadratic tetrahedron
27 Jacobian, and conversion of the apparent density to modulus when integrating over the Young's modulus field. These issues
28 caused up to 2 GPa error in the modulus assignment. For these reasons, we recommend users upgrade to the most recent
29 release of Bonemat.

30 Processing speeds were assessed for the three different element types. Our Python package took significantly longer (110
31 s on average) to perform the calculations compared with the Bonemat software (10 s). Nevertheless, the workflow
32 advantages of the package and added functionality makes 'py_bonemat_abaqus' a useful tool for ABAQUS users.

33

34 **1 Introduction**

35 Bone has heterogeneous material properties, and so in order to create representative finite element models it is essential
36 that the properties are correctly described in order to draw useful results. It is possible to calculate the apparent density of
37 bone from a Computed Tomography (CT) scan. Once the density is known, it can be used to calculate the Young's modulus
38 value, typically using a power equation. Opinion varies as to which mathematical relationships are the most appropriate to
39 use. Helgason *et al.* have published a comprehensive review on the area (Helgason, Perilli et al., 2008) and it is not the
40 purpose of this study to go into further detail on the subject.

41 What we are concerned with is the practical implementation of such mathematical relationships when using the finite
42 element software ABAQUS (Simulia, Dassault Systèmes, Paris, France). It is possible to make such material assignments for
43 an ABAQUS input file using commercial segmentation software, such as Simpleware (Simpleware Ltd., Exeter, UK) or Mimics
44 (Materialise, Leuven, Belgium), but the cost of such software can be prohibitive. Researchers at the Istituto Ortopedico
45 Rizzoli in Bologna, Italy, have created a software program called Bonemat to help with accurate material assignment for a
46 finite element mesh. Bonemat is publically available and can be freely downloaded, making it a useful research tool.
47 However, it has three main limitations:

- 48 1. Bonemat will only work on Microsoft Windows, which can be a problem for users of other operating systems.
- 49 2. Bonemat is not currently compatible with ABAQUS models, so users have to convert their finite element meshes
50 into a compatible format, using an intermediate software such as Hypermesh (Altair Hyperworks, MI, USA) or
51 custom scripting, before material properties can be assigned.
- 52 3. All model information (aside from the mesh) is lost after material assignment; such as, boundary conditions,
53 element sets, or contact definitions. Redefining the element sets and model parameters can add significant time
54 to a project involving multiple models.

55 The algorithm used by Bonemat to calculate the material properties of bone from a CT scan has evolved over the years. The
56 first iteration (V1) calculated the average Hounsfield Unit (HU) value of all voxels found within an element; these were then
57 converted to the modulus of bone to use for the element material (Zannoni, Mantovani et al., 1999). The second iteration
58 (V2) used trilinear interpolation of the CT scan to more accurately estimate the HU value for each element node, and then

59 used numerical integration to compute the HU for the element volume, which was then converted to modulus (Taddei,
60 Pancanti et al., 2004). The third iteration (V3) first converted the CT scan voxels to modulus values, and then performed the
61 linear interpolation and numerical integration to find the modulus for each element volume. Taddei *et al.* demonstrated
62 V2 and V3 algorithms do result in different finite element model results. Strain calculated from models of a femur processed
63 with V2 and V3 algorithms were compared to strains measured experimentally. V3 results had a higher regression
64 coefficient with the experimentally measured strains (0.79) compared with V2 results (0.69) (Taddei, Schileo et al., 2007).

65 We have created an open-source software which has been written in Python (Python Software Foundation, NH, USA) and
66 aims to address the limitations of Bonemat for ABAQUS users. Python was an obvious choice for the program as it is the
67 scripting language used by ABAQUS; furthermore Python works with all operating systems, so the package is multi-platform.
68 It applies the same calculations as Bonemat, allows the same user options, but the mesh data input format is an ABAQUS
69 input file. Furthermore, the program does not remove or change any model parameters already defined in the input file,
70 such as boundary conditions, element sets, or contact definitions. The program also has the added functionality that it can
71 cope with multiple parts, and can 'ignore' parts which are not bone and so do not need material assignment.

72 The aims of this study were to: (1) assess equivalence between material assignment for individual elements using
73 Bonemat3.1 software (current version, Crimi, 2015), Bonemat3.0 software (previous version, Chiarini, 2006) and our Python
74 package ('py_bonemat_abaqus'), (2) assess equivalence for the modulus grouping and element assignment for a whole
75 bone, and (3) compare the speed of the two software packages.

76 **2 Methodology**

77 Material assignment is applied to a finite element mesh in two stages: (1) the software calculates the modulus for each
78 individual element, then (2) the modulus values for the entire mesh are grouped into bins to reduce computation time for
79 the finite element model. Equivalence was assessed separately for these two stages of the material assignment. First, the
80 numerical integration algorithm was checked on individual elements of controlled mesh quality. Second, modulus values
81 were calculated for a CT scan of a hemi-pelvis using the different software packages; this enabled the grouping algorithms
82 to be compared, and was representative of a typical study case. Lastly, the time taken for each software package to perform
83 the calculations on the hemi-pelvis meshes was assessed.

84 **2.1 *Creation of the 'py_bonemat_abaqus' Python package***

85 All scripts were written to be compatible with Python version 2.6 and 2.7 (Python Software Foundation, www.python.org)
86 and were dependent on two other open-source packages: 'numpy' and 'pydicom'. The package can be installed from the
87 Python Package Index using 'pip' or 'easy-install', or can be downloaded and manually installed using the setup.py script
88 (Pegg, 2015). Once installed, the user can either run the script from the command line, or include the statement 'import
89 py_bonemat_abaqus' in their python scripts, to process their ABAQUS input files.

90 **2.2 *Numerical integration accuracy and modulus assignment***

91 The CT scan used was that of a pelvis downloaded from the VAKHUM database, created as part of a project funded by the
92 European Commission under the Information Society Technologies Programme. The dataset was provided by the
93 Laboratory of Human Anatomy and Embryology, University of Brussels (UBL), Belgium (Jan, 2005).

94 Ten single element input files were created with linear tetrahedral (C3D4), quadratic tetrahedral (C3D10), and linear
95 hexahedral (C3D8) elements, and randomly assigned nodal co-ordinates within the CT scan volume. All elements had a
96 Jacobian determinant greater than 0.2; this limit was recommended by Burkhart *et al.* for biomechanical studies of bone
97 (Burkhart, Andrews et al., 2013). The parameters used for the equivalence tests are summarised in Table 1, and described
98 schematically in Figure 1. The parameters used to convert HU to apparent density (ρ_{app}) (Equation 1) and from ρ_{app} to elastic
99 modulus (Equation 2) were based on reported values for the pelvis (Anderson, Peters et al., 2005). Numerical integration
100 across both the HU field (V2) and the modulus field (V3) were compared.

$$101 \quad \rho_{app} = -0.021075 + 0.000786 \text{ HU} \quad \text{Equation 1}$$

$$102 \quad E = 2.0173 \rho_{app}^{2.46} \quad \text{Equation 2}$$

103 The single element meshes were analysed using our 'py_bonemat_abaqus' Python package, Bonemat3.0 downloaded from
104 the BiomedTown website (Chiarini, 2006), and Bonemat3.1 from www.bonemat.org (Crimi, 2015).

105 **2.3 *Modulus grouping and assignment of multiple elements***

106 The hexahedral finite element mesh of the left hemi-pelvis from the VAKHUM database was used for the material
107 assignment of the bone. The linear and tetrahedral meshes were created from the hexahedral mesh by dividing each
108 hexahedron into five tetrahedral elements. The hexahedral element mesh had 5,929 elements, the linear tetrahedral mesh
109 had 29,645 elements, and the quadratic tetrahedral mesh had 29,645 elements. Each model was analysed using the
110 parameters detailed in Table 1. Results were processed using both the V2 and the V3 algorithms.

111 **2.4 *Speed comparison***

112 Python is an interpreted programming language, and therefore scripts written in Python are typically slower compared with
113 pre-compiled programs. For this reason, it was important to quantify any difference in speed. The time taken to analyse
114 the models described in Section 2.3 using the different software packages was assessed, each model was measured 5 times.
115 Tests were performed on a Windows 7 PC with a 64-bit operating system, with four CPUs, 8 GB of RAM, and an Intel Core
116 i5-3470 processor.

117 **3 Results**

118 The assigned modulus values for individual elements were very similar between the newest release of Bonemat (Crimi,
119 2015) and our python package (Figures 2a-c). The mean absolute difference was -0.47 kPa (standard deviation: 13.55 kPa,
120 range: -35.00 to 45.28 kPa). Modulus values from the older release of Bonemat were also similar for the linear hexahedral
121 and tetrahedral elements, with a mean absolute difference of -6.00 kPa (standard deviation: 16.67 kPa, range: -49.00 to
122 14.33 kPa). However, for the quadratic tetrahedral elements, the mean absolute difference was -0.19 GPa (standard
123 deviation: 1.43 GPa, range: -3.01 to 1.57 GPa) (Figure 2c).

124 Modulus assignment of the hemi-pelvis mesh (Figure 3) using the py_bonemat_abaqus software had a good agreement
125 with that calculated by the newest release of Bonemat (Figure 4, Table2). The mean absolute difference was -0.05 kPa
126 (range:-19.48 to 10.23 kPa, standard deviation: 0.62 kPa) (Table 2). The difference in modulus assignment was
127 approximately the same for both numerical integration methods (V2 or V3).

128 Conversely, the modulus assignment with the old version of Bonemat had a mean absolute difference 38.62 MPa
129 (range: -2193.52 to 1907.02 MPa) (Figure 5, Table 3). Furthermore, when using the V3 algorithm the error linearly increased

130 with modulus, which for the CT dataset investigated in this study introduced a maximum error of 2.19 GPa. Examination of
131 the results revealed that the equation to convert apparent density to modulus was not correctly applied when the
132 integration over modulus was selected in the software, which introduced the systematic error.

133 The Python package took longer to perform the calculations compared to both the releases of Bonemat (Figure 6). The
134 python package took a similar time to run for all element types, and this was between 109 and 126 s. The speed of the old
135 version of Bonemat (3.0) was dependent on the element type, ranging from 78s for a hexahedral mesh, to 102s for a
136 quadratic tetrahedral mesh. The newest version of Bonemat (3.1) was the fastest for all element types; the maximum time
137 taken was 20 s to process the quadratic tetrahedral mesh.

138 **4 Discussion**

139 The modulus values calculated by our python package closely matched those from the newest release of Bonemat (Crimi,
140 2015), and any differences were likely to be due to rounding effects. The average difference calculated for the whole CT
141 volume was -0.05 kPa, which to put into context is 0.00000002% of the typical modulus of cortical bone (20.7 GPa), and
142 0.00000003% of the modulus of trabecular bone (14.8 GPa) (Rho, Ashman et al., 1993).

143 Bonemat3.0 had an error in the quadratic tetrahedron modulus calculation which resulted in large differences in modulus
144 assignment. It was possible to reproduce erroneous results by modifying the Jacobian calculation; the two different Jacobian
145 equations are provided as supplementary information. When the representative study case of the hemi-pelvis was analysed
146 for quadratic tetrahedral elements, this calculation error was found to introduce an error of up to 2 GPa, which could have
147 a significant impact on study results. A second bug was observed in the old version of Bonemat software for the V3
148 calculation, where the equation to convert apparent density to modulus was not correctly applied and introduced additional
149 errors. These results call into question the accuracy of published studies which have used Bonemat3.0 to assign materials
150 to quadratic tetrahedral elements. We recommend that users update their Bonemat software to the newest version which
151 has addressed these issues.

152 The python package 'py_bonemat_abaqus' took 6 to 24 times longer to process the models, depending on the element
153 type, compared to the newest release of Bonemat. This increased run time may be an issue for extremely large models or
154 high resolution CT datasets, however, the research time that will be saved by users not having to convert the file format of

155 the mesh, or perform manual edits to the input file to re-assign element sets prior to solving the model, may offset the
156 increased time to assign the material properties.

157 In summary, the 'py_bonemat_abaqus' Python package produced equivalent results to Bonemat3.1 (the current release),
158 but not to Bonemat3.0 (the previous version from the BiomedTown website) which contained some calculation errors. The
159 'py_bonemat_abaqus' Python package was slower to process the models, but will provide ABAQUS users with a useful
160 method to assign material properties of bone to finite element models, with an easier workflow, fewer limitations, and as
161 it is written in Python, it can be directly incorporated into scripts written to interact with ABAQUS. Through collaboration
162 with the Istituto Ortopedico Rizzoli in Bologna, work is underway to incorporate the ABAQUS input and output parts of the
163 'py_bonemat_abaqus' script into Bonemat.

164 **Acknowledgements**

165 The methodology and general algorithm for the Python package was based upon the original Bonemat software and we
166 would like to thank the Istituto Ortopedico Rizzoli in Bologna, Italy for their work. Furthermore, the CT scan and finite
167 element mesh used for the testing was from the VAKHUM database, which was kindly provided by the Laboratory of Human
168 Anatomy and Embryology, University of Brussels (ULB) in Belgium and the Istituto Ortopaedico Rizzoli. We would also like
169 to thank the Python Software Foundation, and the Python Package Index, for enabling the source code to be hosted online
170 so other researchers can download and use the 'py_bonemat_abaqus' package.

171 **Conflict of Interest**

172 The authors did not receive payment or services from any third party for any aspect of the submitted work. The authors
173 have received support for research projects which were unrelated to the present study.

174 **References**

- 175 Anderson, A. E., Peters, C. L., Tuttle, B. D., Weiss, J. A., 2005. Subject-Specific Finite Element Model of the Pelvis:
176 Development, Validation and Sensitivity Studies. *Journal of Biomechanical Engineering* 127, 364-373.
177 Burkhart, T. A., Andrews, D. M., Dunning, C. E., 2013. Finite element modeling mesh quality, energy balance and validation
178 methods: A review with recommendations associated with the modeling of bone tissue. *Journal of Biomechanics*
179 46, 1477-1488.
180 Chiarini, A., 2006. Download Bonemat Software. Retrieved 6/8/2015, 2015, from
181 https://www.biomedtown.org/biomed_town/B3C_Building/products/bonemat/download_html/.
182 Crimi, G., 2015. Bonemat. Retrieved 6/8/2015, 2015, from <http://www.bonemat.org/downloads.php>.

183 Helgason, B., Perilli, E., Schileo, E., Taddei, F., Brynjólfsson, S., Viceconti, M., 2008. Mathematical relationships between
 184 bone density and mechanical properties: A literature review. *Clinical Biomechanics* 23, 135-146.
 185 Jan, S. V. S., 2005. The VAKHUM Project: Virtual Animation of the Kinematics of the Human. *Theoretical Issues in*
 186 *Ergonomics Science* 6.3-4, 277-279.
 187 Pegg, E., 2016. py_bonemat_abaqus.. Retreved 7/1/2016, 2016, from
 188 https://pypi.python.org/pypi/py_bonemat_abaqus/
 189 Rho, J. Y., Ashman, R. B., Turner, C. .H., 1993. Young's modulus of trabecular and cortical bone material: ultrasonic and
 190 microtensile measurements. *Journal of Biomechanics* 26, 111-119.
 191 Taddei, F., Pancanti, A., Viceconti, M., 2004. An improved method for the automatic mapping of computed tomography
 192 numbers onto finite element models. *Medical Engineering and Physics* 26, 61-69.
 193 Taddei, F., Schileo, E., Helgason, B., Christofolini, L., Viceconti, M., 2007. The material mapping strategy influences the
 194 accuracy of CT-based finite element models of bones: An evaluation against experimental measurements.
 195 *Medical Engineering and Physics* 29, 973-979.
 196 Zannoni, C., Mantovani, R., Viceconti, M., 1999. Material properties assignment to finite element models of bone
 197 structures: a new method. *Medical Engineering and Physics* 20, 735-740.

198 **Table and Figure Captions**

199 Table 1. Parameters used when assessing equivalence between material assignments of the software packages.

200 Table 2. Summary of the modulus difference (py_bonemat_abaqus – Bonemat3.1) of the hemi-pelvis when analysed using
 201 different elements and numerical integration algorithms. Results are shown in kPa.

202 Table 3. Summary of the modulus difference (py_bonemat_abaqus – Bonemat3.0) of the hemi-pelvis when analysed using
 203 different elements and numerical integration algorithms. Results are shown in MPa.

204 Figure 1. Schematic illustration of the parameters detailed in Table 1 and how they influence the grouping of the modulus
 205 results.

206 Figure 2. Comparison of the modulus values output by Bonemat3.1 and Bonemat3.0 to the py_bonemat_abaqus python
 207 package, for the (a) linear hexahedron, (b) linear tetrahedron, and (c) quadratic tetrahedron (n=10) with the V2 algorithm.

208 Figure 3. Comparison of the modulus values assigned to the hexahedral mesh of the hemi-pelvis by py_bonemat_abaqus
 209 and Bonemat 3.1 using the V3 algorithm. An anterior-posterior view of the whole mesh is shown, and a cross-sectional view
 210 through the volume centroid.

211 Figure 4. Bland Altman plots summarising the difference in assigned modulus (in GPa) for the hemi-pelvis mesh for all
 212 element types combined. Differences shown compare the py_bonemat_abaqus to Bonemat 3.1 analysed with (a) the V2
 213 algorithm, (b) the V3 algorithm. Bland-Altman plots with kPa y-axis scales are provided as supplementary information

214 Figure 5. Bland Altman plots summarising the difference in assigned modulus (in GPa) for the hemi-pelvis mesh for all
 215 element types combined. Differences shown compare the py_bonemat_abaqus to Bonemat 3.0 analysed with (a) the V2
 216 algorithm, (b) the V3 algorithm.

217 Figure 6. Summary of the time taken for each software to perform the calculations and output the modified mesh file for
 218 the three element types. Error bars represent the standard deviation in the time measurement (n=5). Tests were
 219 performed on a Windows 7 PC with a 64-bit operating system, with four CPUs, 8 GB of RAM, and an Intel Core i5-3470
 220 processor.

221

Parameter	Description	Set value
Gap Value	Modulus interval used when grouping the modulus values	0.01 GPa
Integration order	Order of the numerical integration used across the element	4
CT calibration coefficients [$\rho_{app}=a+b \text{ HU}$]	Calibration parameters a and b of the equation used to calculate the apparent density of bone (ρ_{app}) from the CT Hounsfield Unit (HU)	$a = -0.021075$, $b = 0.000786$
Calibration correction	Option which allows correction of the calibration with up to three linear correlation.	Not applied
Modulus calculation parameters [$E=a+b \rho_{app}^c$]	Parameters a , b and c of the power equation used to convert the apparent density of bone to Young's modulus	$a = 0$, $b = 2.0173$, $c = 2.46$
Minimum modulus value	Any modulus values below the minimum are changed to the minimum value	0.000001

223 Table 1. Parameters used when assessing equivalence between material assignments of the software packages.

Element Type	Algorithm	Mean Absolute Difference	Standard Deviation	Minimum	Maximum
Linear Hex	V2	-0.11	0.96	-18.38	6.91
Linear Tet	V2	-0.18	0.09	-0.24	0.05
Quad Tet	V2	0.00	0.13	-6.15	4.27
All Elements	V2	-0.10	0.39	-18.38	6.91
Linear Hex	V3	-0.07	0.92	-19.48	10.23
Linear Tet	V3	0.05	1.47	-10.01	0.45
Quad Tet	V3	0.00	0.16	-6.90	5.11
All Elements	V3	-0.01	0.85	-9.48	10.23
All results combined		-0.05	0.62	-19.48	10.23

224 Table 2. Summary of the modulus difference (py_bonemat_abaqus – Bonemat3.1) of the hemi-pelvis when analysed using
225 different elements and numerical integration algorithms. Results are shown in kPa.

Element Type	Algorithm	Mean Absolute Difference	Standard Deviation	Minimum	Maximum
Linear Hex	V2	0.00	0.00	-0.04	0.02
Linear Tet	V2	0.00	0.08	-0.14	10.00
Quad Tet	V2	-0.21	36.58	-2193.52	1242.79
All Elements	V2	-0.07	12.22	-2193.52	1242.79
Linear Hex	V3	81.06	108.60	5.70	672.33
Linear Tet	V3	75.68	121.16	-2.38	812.07
Quad Tet	V3	75.18	124.04	-1226.69	1907.02
All Elements	V3	77.31	117.93	-1226.69	1907.02
All results combined		38.62	65.08	-2193.52	1907.02

226 Table 3. Summary of the modulus difference (py_bonemat_abaqus – Bonemat3.0) of the hemi-pelvis when analysed using
227 different elements and numerical integration algorithms. Results are shown in MPa.

Figure 1. Schematic illustration of the parameters detailed in Table 1 and how they influence the grouping of the modulus results.

Figure 2. Comparison of the modulus values output by Bonemat3.1 and Bonemat3.0 to the py_bonemat_abaqus python package, for the (a) linear hexahedron, (b) linear tetrahedron, and (c) quadratic tetrahedron ($n=10$) with the V2 algorithm.

235

236 Figure 3. Comparison of the modulus values assigned to the hexahedral mesh of the hemi-pelvis by py_bonemat_abaqus
 237 and Bonemat 3.1 using the V3 algorithm. An anterior-posterior view of the whole mesh is shown, and a cross-sectional view
 238 through the volume centroid.

239

Figure 4. Bland Altman plots summarising the difference in assigned modulus (in GPa) for the hemi-pelvis mesh for all element types combined. Differences shown compare the py_bonemat_abaqus to Bonemat 3.1 analysed with (a) the V2 algorithm, (b) the V3 algorithm. Bland-Altman plots with kPa y-axis scales are provided as supplementary information

Figure 5. Bland Altman plots summarising the difference in assigned modulus (in GPa) for the hemi-pelvis mesh for all element types combined. Differences shown compare the py_bonemat_abaqus to Bonemat 3.0 analysed with (a) the V2 algorithm, (b) the V3 algorithm.

Figure 6. Summary of the time taken for each software to perform the calculations and output the modified mesh file for the three element types. Error bars represent the standard system deviation in the time measurement (n=5). Tests were performed on a Windows 7 PC with a 64-bit operating system, with four CPUs, 8 GB of RAM, and an Intel Core i5-3470 processor.

254 **Supplementary Information 1**

255 *Jacobian calculation for a quadratic tetrahedron by the 'py_bonemat_abaqus' Python package.*

256 Assuming the shape functions defined in Equation 1, where l , r , s , and t represent the natural co-ordinate system of a
 257 tetrahedron, the differentiation with respect to each of the natural co-ordinates is shown in Equations 2-5. Based on these
 258 differentiated shape functions, the Jacobian can be calculated using the natural co-ordinates and the nodal co-ordinates
 259 $[(x^1, y^1, z^1), (x^2, y^2, z^2) \dots (x^{10}, y^{10}, z^{10})]$ using Equation 6 (the more compact transpose matrix is shown).

260

$$261 \quad N = \begin{bmatrix} (2l-1)l \\ (2r-1)r \\ (2s-1)s \\ (2t-1)t \\ 4lr \\ 4rs \\ 4ls \\ 4lt \\ 4rt \\ 4st \end{bmatrix} \quad (1)$$

$$262 \quad \frac{\delta N}{\delta l} = \begin{bmatrix} 4l-1 \\ 0 \\ 0 \\ 0 \\ 4r \\ 0 \\ 4s \\ 4t \\ 0 \\ 0 \end{bmatrix} \quad \frac{\delta N}{\delta r} = \begin{bmatrix} 0 \\ 4r-1 \\ 0 \\ 0 \\ 4l \\ 4s \\ 0 \\ 0 \\ 4t \\ 0 \end{bmatrix} \quad \frac{\delta N}{\delta s} = \begin{bmatrix} 0 \\ 0 \\ 4s-1 \\ 0 \\ 0 \\ 4r \\ 4l \\ 0 \\ 0 \\ 4t \end{bmatrix} \quad \frac{\delta N}{\delta t} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 4t-1 \\ 0 \\ 0 \\ 0 \\ 4l \\ 4r \\ 4s \end{bmatrix} \quad (2), (3), (4), (5)$$

$$263 \quad J^T = 4 \begin{bmatrix} \frac{1}{4} & x^1 \left(l - \frac{1}{4} \right) + x^5 r + x^7 s + x^8 t & y^1 \left(l - \frac{1}{4} \right) + y^5 r + y^7 s + y^8 t & z^1 \left(l - \frac{1}{4} \right) + z^5 r + z^7 s + z^8 t \\ \frac{1}{4} & x^2 \left(r - \frac{1}{4} \right) + x^5 l + x^6 s + x^9 t & y^2 \left(r - \frac{1}{4} \right) + y^5 l + y^6 s + y^9 t & z^2 \left(r - \frac{1}{4} \right) + z^5 l + z^6 s + z^9 t \\ \frac{1}{4} & x^3 \left(s - \frac{1}{4} \right) + x^6 r + x^7 l + x^{10} t & y^3 \left(s - \frac{1}{4} \right) + y^6 r + y^7 l + y^{10} t & z^3 \left(s - \frac{1}{4} \right) + z^6 r + z^7 l + z^{10} t \\ \frac{1}{4} & x^4 \left(t - \frac{1}{4} \right) + x^8 l + x^9 r + x^{10} s & y^4 \left(t - \frac{1}{4} \right) + y^8 l + y^9 r + y^{10} s & z^4 \left(t - \frac{1}{4} \right) + z^8 l + z^9 r + z^{10} s \end{bmatrix}^T \quad (6)$$

264 *Jacobian calculation used by the old release of the Bonemat software*

265 The shape functions used for the old release of Bonemat software (Chiarini, 2006) were the same, but l was defined as
 266 equal to $1-r-s-t$ and then a 3x3 Jacobian matrix was used (Equation 7). This Jacobian calculation was erroneous.

267

$$268 \quad J = \begin{bmatrix} x^1(1-4l) + x^2(4r-1) + x^5(4l-4r) + (x^6-x^7)4s + (x^9-x^8)4t & \dots \text{same for } y & \dots \text{same for } z \\ x^1(1-4l) + x^3(4s-1) + x^7(4l-4s) + (x^4-x^5)4r + (x^{10}-x^8)4t & \dots \text{same for } y & \dots \text{same for } z \\ x^1(1-4l) + x^4(4t-1) + x^8(4l-4t) + (x^9-x^5)4r + (x^{10}-x^7)4s & \dots \text{same for } y & \dots \text{same for } z \end{bmatrix} \quad (7)$$

269

270 **Supplementary Information 2 – Individual Bland Altman Plots**

271
272 Table S1. Bland Altman plots comparing the difference in modulus assignment with Bonemat3.1 and the
273 py_bonemat_abaqus package (py_bonemat_abaqus - Bonemat3.1) of an element mesh of the hemi-pelvis mesh. Difference
274 shown is in kPa.

275

276 Table S2. Bland Altman plots comparing the difference in modulus assignment with Bonemat3.0 and the
 277 py_bonemat_abaqus package (py_bonemat_abaqus - Bonemat3.0) of an element mesh of the hemi-pelvis mesh. Difference
 278 shown is in GPa.