# Computationally-Optimized Bone Mechanical Modeling from High-Resolution Structural Images

**Jeremy F. Magland*, Ning Zhang, Chamith S. Rajapakse, Felix W. Wehrli**

Laboratory for Structural NMR Imaging, Department of Radiology, University of Pennsylvania Medical Center, Philadelphia, Pennsylvania, United States of America

## Abstract

Image-based mechanical modeling of the complex micro-structure of human bone has shown promise as a non-invasive method for characterizing bone strength and fracture risk *in vivo*. In particular, elastic moduli obtained from image-derived micro-finite element (μFE) simulations have been shown to correlate well with results obtained by mechanical testing of cadaveric bone. However, most existing large-scale finite-element simulation programs require significant computing resources, which hamper their use in common laboratory and clinical environments. In this work, we theoretically derive and computationally evaluate the resources needed to perform such simulations (in terms of computer memory and computation time), which are dependent on the number of finite elements in the image-derived bone model. A detailed description of our approach is provided, which is specifically optimized for μFE modeling of the complex three-dimensional architecture of trabecular bone. Our implementation includes domain decomposition for parallel computing, a novel stopping criterion, and a system for speeding up convergence by pre-iterating on coarser grids. The performance of the system is demonstrated on a dual quad-core Xeon 3.16 GHz CPUs equipped with 40 GB of RAM. Models of distal tibia derived from 3D in-vivo MR images in a patient comprising 200,000 elements required less than 30 seconds to converge (and 40 MB RAM). To illustrate the system's potential for large-scale μFE simulations, axial stiffness was estimated from high-resolution micro-CT images of a voxel array of 90 million elements comprising the human proximal femur in seven hours CPU time. In conclusion, the system described should enable image-based finite-element bone simulations in practical computation times on high-end desktop computers with applications to laboratory studies and clinical imaging.

**Citation:** Magland JF, Zhang N, Rajapakse CS, Wehrli FW (2012) Computationally-Optimized Bone Mechanical Modeling from High-Resolution Structural Images. PLoS ONE 7(4): e35525. doi:10.1371/journal.pone.0035525

**Editor:** Xiaoyuan Chen, NIH, United States of America

**Received** December 23, 2011; **Accepted** March 20, 2012; **Published** April 25, 2012

**Funding:** This work was supported by National Institutes of Health grants R01 AR55647, R01 AR55356 and K25 AR 060283. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

**Competing Interests:** The authors have declared that no competing interests exist.

* E-mail: Jeremy.Magland@gmail.com

## Introduction

Large-scale finite-element simulations of complex physical systems (e.g. involving 10 million or more finite elements) are being used increasingly in many areas of science, engineering, biomedical and clinical research and industry [1,2,3,4]. However, most existing large-scale finite-element simulation programs require significant computing resources, which may hamper their use in common laboratory and clinical environments. The development of computationally efficient finite-element solvers for targeted applications is therefore of great interest.

Image-based micro-finite-element (μFE) modeling on the basis of high-resolution medical images has shown promise as a technique for mechanical characterization of the complex micro-structure of bone. Both magnetic resonance (MR) and peripheral high-resolution computed tomography (HR-pQCT) have already demonstrated the ability to monitor alterations in bone mechanical properties resulting from disease progression or drug intervention [5,6] or for assessment of fracture risk [7,8]. FE analyses at multiple scales from macro- to micro-structure have also been proposed as possible means to provide insight into failure mechanisms [9].

Bone is classified into two structural types: cortical and trabecular. Both types of bone remodel throughout human life, with old bone being resorbed and new bone being deposited. Remodeling controls the reshaping or replacement of bone during growth and following injury, and generally occurs in response to changes in functional demands of mechanical loading [10]. Perturbation in bone mineral homeostasis, e.g. due to hormone loss following menopause [11,12] or extended exposure to microgravity [13] causes a remodeling imbalance with greater rate of resorption than new bone formation, resulting in structural and mechanical impairment of the skeleton due to architectural deterioration along with net loss of trabecular and cortical bone [14,15,16]. The above scenario is characteristic of the etiology of osteoporosis, a condition that leads to increased risk of fracture.

High-resolution image-based μFE analysis is able to simulate the effects of mechanical loading of bone, thus providing insight into the relationship between bone microarchitecture and bone strength. Excellent agreement has been noted between biomechanical compression tests and μFE-derived elastic moduli based on images acquired at high spatial resolution [17,18,19]. Unlike direct mechanical testing, the gold standard for determining bone mechanical competence, image-based μFE modeling is nondestructive and is hence feasible *in vivo* [6,20].

Human trabecular bone is a complex network of interconnected plates and struts on the order of 100–150 μm thickness [15] whereas the macroscopic scale of bone is on the order of centimeters or even tens of centimeters. The computational demands (in terms of RAM and CPU) can therefore be enormous for accurate high-resolution simulation of even a portion of the bone (such as the vertebrae, distal radius or proximal femur–locations of high fracture incidence). For example, FE simulation of a single human vertebral body would require around 185 million elements at an element size (and thus image voxel size) of 30 μm. While such resolution is far beyond any in-vivo imaging modality's capability, the potential to predict bone mechanical properties on the basis of lower-resolution in vivo images, is of significant clinical interest [6,20]. Under the best of circumstances *in vivo* MRI and CT currently yield an effective resolution on the order of 100–200 μm at selected skeletal locations [20,21,22] in practical scan times (MRI) and acceptable radiation dose (CT), which typically reduces data size to 10 million elements or less. However, it is conceivable that pre-processing of the images to higher apparent resolution via interpolation techniques such as subvoxel processing [23] or zero filling in Fourier space [24] may significantly enhance accuracy in the prediction of the bone's mechanical behavior, but would also significantly increase data array size. Although not addressed in this paper, simulations in the nonlinear regime increase computational demands by an order of magnitude or more, and are thus impractical unless computational efficiency is substantially augmented.

Here, we investigate the feasibility of large-scale FE simulations (performed on desktop personal computers) and describe an optimized FE solver designed for high-resolution image-based computational bone mechanics of systems with 10–100 million elements within the constraints of standard workstations in minutes to hours. These advances are achieved through algorithmic improvements involving effective memory usage, accelerated convergence and parallelization. Critical to these endeavors is a reduction in the number of iterations required toward convergence of the solution. We show that this goal can be achieved by starting iteration on coarser grids (i.e. using larger size thereby reducing the number of elements). Further we describe an iteration procedure that enables a more effective estimate of the relative error in total stress, accurately indicating when to halt the conjugate gradient iterations. Lastly, we show that significant speed enhancements can be achieved by making efficient usage of the available processors through parallelization of the computing tasks. The performance of the FE solver is illustrated with applications to human specimen micro-CT and in vivo high-resolution MR images as input into the model to estimate stiffness and failure load.

## Materials and Methods

### Definition of the linear system

Image-based estimation of macroscopic mechanical properties of bone involves (a) defining the image-derived structural bone model (b) simulating the induced macroscopic strain by applying appropriate boundary conditions, (c) solving for the resulting equilibrium displacements throughout the structure, and (d) computing stiffness from macroscopic stress/strain ratios. In the linear elastic regime (which we assume throughout), local stress and strain are linearly related by Hooke's Law:

$$\sigma = C\varepsilon, \tag{1}$$

where

$$\sigma = \begin{bmatrix} \sigma_{xx} & \sigma_{yy} & \sigma_{zz} & \sigma_{yz} & \sigma_{zx} & \sigma_{xy} \end{bmatrix}^T$$

and

$$\varepsilon = \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{yy} & \varepsilon_{zz} & \varepsilon_{yz} & \varepsilon_{zx} & \varepsilon_{xy} \end{bmatrix}^T$$

are the local stress and strain vectors, respectively, and $C = (C_{ij})$ is the $6 \times 6$ stiffness matrix for the material. In the case of isotropic material, the stiffness matrix takes the form

$$C = \frac{E}{(1-v)(1-2v)} \begin{bmatrix} 1-v & v & v & 0 & 0 & 0 \\ v & 1-v & v & 0 & 0 & 0 \\ v & v & 1-v & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2v}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2v}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2v}{2} \end{bmatrix}, \tag{2}$$

where $E$ is Young's modulus and $v$ is Poisson's ratio. Here, for trabecular bone, we assume $v = 0.3$, and $E = 15\,\mathrm{GPa}\cdot BVF$, where $BVF$ represents the voxel-wise bone- volume fraction (see [25] for MR or [26] for CT). Even though on a microscopic scale the bone material modulus is not isotropic [27,28], for most applications the assumption of voxel-wise isotropy is warranted.

Following [29], we let each 3D image voxel represent a single hexahedral (brick) element in our finite element model. By assuming a tri-linear displacement field within each brick element, the microscopic Hooke's law defines a linear relationship between the vertex displacements $\vec{u}_{i,j,k}^{(B)}$ and the induced vertex forces $\vec{F}_{i,j,k}^{(B)}$(*i,j,k = 0,1* are the coordinate indices). The superscript "(*B*)" indicates that this is the force acting on the vertex by a single element, and is therefore only one component of the total force at the vertex. The method for determining the precise relationship between induced vertex forces and displacements for a single element is described in Appendix S1. This relationship can be expressed as

$$\vec{F}^{(B)} = C_{\mathrm{kernel}} \vec{u}^{(B)}, \tag{3}$$

where $C_{kernel}$ is a $24 \times 24$ kernel matrix. The total force at a vertex $v$ in the direction $d$, denoted by $F(v,d)$, is the sum of all forces at $v$ (in direction d) induced by the brick elements containing $v$.

Simulation of applied strain (step (b) above) involves application of boundary conditions of the form:

$$u(v,d) = c_{v,d}, \tag{4}$$

where $u(v,d)$ is the displacement of the vertex $v$ in the direction $d$ at selected vertex locations (usually on a boundary surface of the image volume). The condition of force equilibrium (step (c)) can then be expressed as

$$F(v,d) = 0, \tag{5}$$

at all free vertex/direction pairs (i.e., those without applied boundary conditions). Therefore, the number of equations in this linear system is equal to $N_v$, the number of free (displacement) variables. The (macroscopic) linear system for the force can be expressed as

$$AU = B, \tag{6}$$

where $A$ is a sparse $N_v \times N_v$ matrix referred to as the *macroscopic stiffness matrix* and $U$ are the displacements at all free vertex/direction pairs. The right-hand side, B is defined according to the applied boundary conditions. The central (and most time-consuming) step in the mechanical modeling procedure is to solve this linear system (6).

## Conjugate gradient iteration

Because the microscopic stiffness matrix $C$ (e.g. Equation (2)) is symmetric, one can show that the macroscopic stiffness matrix $A$ is positive definite (this condition is equivalent to the total energy always being nonnegative, for any vertex displacement configuration). We may therefore use the preconditioned conjugate-gradient (PCG) algorithm to solve equation (6) [30,31]. Because we will subsequently refer to the details of this algorithm, the processing steps are outlined below for the conjugate gradient procedure.

### Algorithm 1. Conjugate Gradient Algorithm

**Step 0**: Select an initial displacement configuration $u_0$, compute the residual $r_0$ and set the initial search direction (for simplicity we leave out the preconditioner in this description):

$$r_0 = B - Au_0$$

$$p_0 = r_0.$$

Set $n := 0$.

**Step 1**: Compute $q_n = Ap_n$.

**Step 2**: Compute $\alpha_n = \frac{\langle r_n, r_n \rangle}{\langle p_n, q_n \rangle}$ (where $\langle a, b \rangle$ stands for inner product of two vectors $a$, $b$), and then compute the new displacement and residual vectors:

$$u_{n+1} = u_n + \alpha_n p_n$$

$$r_{n+1} = r_n - \alpha_n q_n$$

**Step 3**: Compute $\beta_n = \frac{\langle r_{n+1}, r_{n+1} \rangle}{\langle r_n, r_n \rangle}$ and then compute the new search direction:

$$p_{n+1} = \beta_n p_n + r_{n+1}.$$

**Step 4**: If within error tolerance, stop. Otherwise increment $n := n+1$ and return to Step 1.

## Memory usage estimation

Even when using a memory-efficient sparse-matrix storage scheme, construction of the $N_v \times N_v$ sparse matrix $A$ is highly memory-intensive as compared with the element-by-element (EBE) approach [32,33]. Because each vertex has up to *27* neighbor vertices (including itself), a single row of the matrix can have up to 81 non-zero entries (three displacement directions for each vertex), requiring storage of up to (81 entries per variable)x(4 bytes per entry) = 324 bytes per free variable (this is even neglecting memory required to store the entry locations). Throughout this paper, we empirically estimate the number of variables as approximately 4 times the number of elements, $N_v \approx 4N_e$, where $N_e$ is the number of elements (note: if bone were to occupy the entire volume of the image, this ratio would equal 3, but accounting for boundary vertices, this ratio tends to be closer to 4). Therefore, the full-matrix method (i.e. storing $A$ using an efficient sparse matrix scheme) requires up to 1,296 bytes per element for matrix storage alone.

In contrast, the EBE method demands substantially less memory, since only the *BVF* scaling factor and vertex indices need to be stored for each element. The key assumption is that the $24 \times 24$ kernel matrix ($C_{kernel}$) is identical (up to BVF scaling factor) at all elements. However, there is a tradeoff in terms of computation time. With the sparse matrix construction method, each matrix multiplication (Step 1 in the CG algorithm) involves up to $81N_v$ multiplication operations, compared with $24 \times 24N_e$ for the EBE approach. Assuming $N_v \approx 4N_e$ as above, this suggests that EBE would be slower by a factor of around 1.8. However, the true ratio for comparing the two methods may differ depending on the efficiency of the sparse matrix multiplication algorithm, and the actual average number of entries per row (recall that 81 is an upper estimate). Nevertheless, the significant memory savings of the EBE technique generally outweighs the modest loss in iteration speed.

Table 1 shows a breakdown of the total theoretical memory usage for the EBE technique. In addition to bone-volume fraction (1 byte per element), 24 variable indices must be stored at each brick element (8 vertices×3 directions per vertex). Assuming 32-bit integers are used for storing indices, this requires $24 \times 4 = 96$ bytes per element. However, if variables are stored sequentially according to 4-dimensional coordinates (three spatial and one direction, with direction as the inner iteration), then only one sixth of these variable indices need to be stored by the element, because other indices can be obtained by offsetting the base indices. Thus only 16 bytes per element are required for variable indices. The bulk of memory usage (≈80 bytes per element) is required by the five CG vectors: displacement (u), residual (r), search direction (p), search direction multiplied by A ($q = Ap$), and the vector storing the Jacobi preconditioner. Finally, we need to store the variable index lookup map, so that vertex-variable indices can be related to locations on the original image grid. This requires 4 bytes per

**Table 1.** Estimated memory usage by number of elements.

| | Expected Memory Usage: Element-by-Element Method |
|---|---|
| Bone-volume fraction map | (1 byte)$\times N_e$ |
| Element vertex-variable indices | (16 bytes)$\times N_e$ |
| Five vectors in the conjugate gradient algorithm: u, r, p, q, +preconditioner | (4 bytes)$\times N_v \times 5 \approx$(80 bytes)$\times N_e$ |
| Variable index lookup map | (4 bytes)$\times N_x N_y N_z \approx$(32 bytes)$\times N_e$ |
| **Total** | **≈(130 bytes)$\times N_e$** |

doi:10.1371/journal.pone.0035525.t001

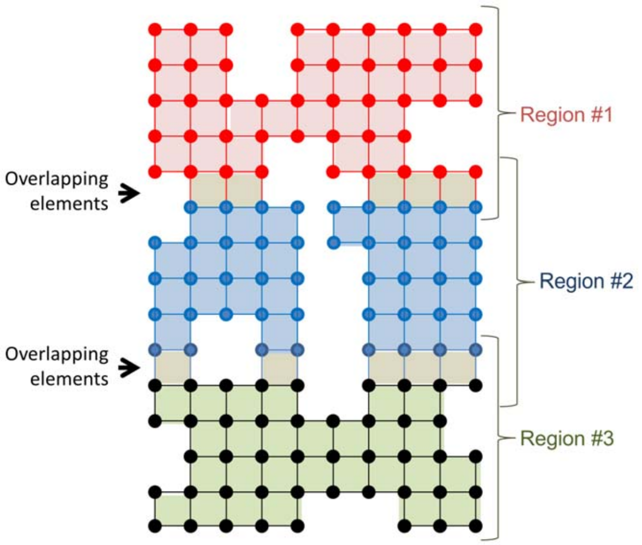

**Figure 1. Two-dimensional representation of the finite element model, with each point representing a vertex or node in the system.** Parallel computing requires additional memory allocation since vertices located at the interfaces between sub-regions must be stored twice.
doi:10.1371/journal.pone.0035525.g001

image vertex, or at most 32 bytes per element (assuming at least 1/8 of image voxels are occupied by positive bone volume fraction).

As described below, parallel computing requires allocation of additional memory since data located at the interface between sub-regions must be duplicated between multiple processor threads (see Figure 1). Assuming that the structure is split into sub-regions along the Z-direction (inferior-superior direction in most cases), the formula for the fraction of overhead is given by $\frac{2(K-1)}{N_z+1}$ where K is the number of threads (or number of sub-regions) and $N_z$ is the number of voxels along the Z direction. For example, if $K=8$ threads are used and $N_z=100$, then the overhead would be around 14%, for an expected memory usage of around **150 bytes per element**.

## Parallel processing

To make use of multiple processors for speeding up the simulation, it is necessary to divide the workload among the processors [34]. Ideally, with perfect distribution of computations, total simulation time will be reduced by a factor equal to the number of processors. However, as with memory usage estimates, we need to consider the cost of additional computations performed on the interfaces between sub-regions. Below is a parallelized version of the PCG algorithm. Within each sub-region, we distinguish between *inner* and *outer* vertices according to the color-coding of Figure 1. For example, the blue vertices are the inner vertices of Region #2, whereas this region also contains two rows of outer vertices (red and black).

Algorithm 2 (below) has been reorganized (compare with Algorithm 1) in order to facilitate parallel computing. Steps 1a, 1b, and 3a can all be implemented independently within each sub-region. On the other hand, steps 2 and 3b involve interaction between sub-regions. Fortunately, 2 and 3b do not require significant computation time as compared with the remainder of the algorithm. Therefore, we can expect to achieve close to a K-fold speedup. Toward this end, the sub-regions should be chosen so that the vertices are distributed as uniformly as possible. In the present implementation, we chose the sub-region bounding planes to be parallel to one another, and optimize their positions so that each sub-region has an approximately equal share of vertices.

## Algorithm 2. Parallelized Conjugate Gradient Algorithm

**Step 0**: Select an initial displacement configuration $u_0$, compute the residual $r_0$ and set the initial search direction (for simplicity we leave out the preconditioner):

$$r_0=B-Au_0$$
$$p_0=r_0.$$

Set $n:=0$.

Divide the volume into $K$ sub-regions $S_1, \ldots, S_k$ (as shown in Figure 1), where K is the number of processing threads. The choice of sub-regions should be load-balancing (e.g. using the technique described below).

**Step 1a**: For each sub-region, compute $q_n=Ap_n$ on the inner vertices.

**Step 1b**: For each sub-region $S_k$, compute the following partial inner products

$$\langle r_n,r_n\rangle_{I_k}, \langle r_n,q_n\rangle_{I_k}, \langle p_n,q_n\rangle_{I_k}, \langle q_n,q_n\rangle_{I_k}$$

where

$$\langle v,w\rangle_{I_k}=\sum_{m\in I_k} v_m w_m$$

And the sum is computed over the inner vertex indices $I_k$ of the subregion $S_k$.

**Step 2**: Compute the full inner-products $\langle r_n,r_n\rangle$, $\langle r_n,q_n\rangle$, $\langle p_n,q_n\rangle$, $\langle q_n,q_n\rangle$ by summing over the partial inner products from step 1b in the following manner:

$$\langle v,w\rangle=\sum_{k=1}^{K}\langle v,w\rangle_{I_k}.$$

Then, compute $\alpha_n=\frac{\langle r_n,r_n\rangle}{\langle p_n,q_n\rangle}$.

$$\beta_n=\frac{\langle r_{n+1},r_{n+1}\rangle}{\langle r_n,r_n\rangle}=1+2\frac{\langle r_n,q_n\rangle}{\langle p_n,q_n\rangle}+\frac{\langle r_n,r_n\rangle\langle q_n,q_n\rangle}{\langle p_n,q_n\rangle^2}.$$

**Step 3a**: For each sub-region, compute the new displacement on all (inner and outer) vertices

$$u_{n+1}=u_n+\alpha_n p_n$$

and compute the new residual and search direction vectors on the inner vertices only:

$$r_{n+1}=r_n-\alpha_n q_n$$
$$p_{n+1}=\beta_n p_n+r_{n+1}.$$

**Step 3b**: For each sub-region, update $p_{n+1}$ on the outer vertices by retrieving the information from the inner vertices of neighboring sub-regions.

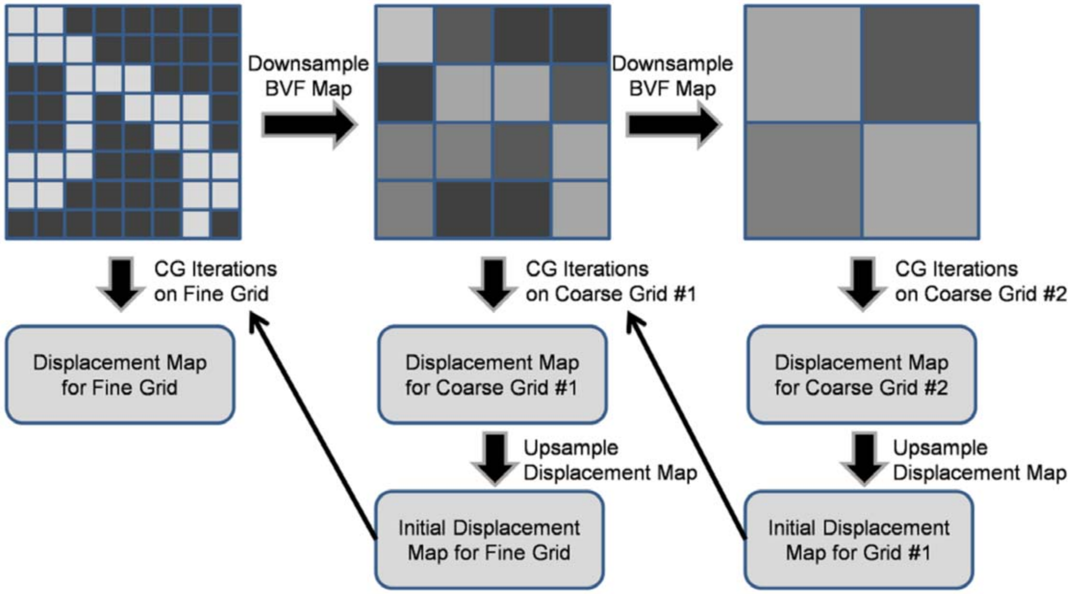

**Figure 2. Total simulation time can be significantly decreased by performing initial iterations on coarser grids prior to solving the system on the original (fine) grid.** Since the solution obtained at the coarser scale is already close to the final solution, fewer iterations are required at the finest scale.

doi:10.1371/journal.pone.0035525.g002

**Step 4**: If within error tolerance, stop. Otherwise increment $n := n+1$ and go to Step 1.

## Convergence criterion

Traditionally, the magnitude of the residual vector $r_n$, is used to determine when to halt the conjugate gradient iteration procedure. However, the direct quantity of interest is the computed total stress (as a function of applied strain), and we are therefore most interested in how closely this computed value at each iteration is to its converged (i.e., "true") value. Here we describe a method for estimating the relative error in this total stress in order to more accurately determine when to halt the conjugate gradient iterations.

Let $S_n$ be the computed total stress after the $n_{th}$ iteration. We assume that, after a finite number of iterations, $S_n$ will converge approximately exponentially to its (unknown) true value $S_\infty$ according to

$$S_n \approx S_\infty + ae^{-bn}.$$

Taking the log of the absolute derivative, with respect to $n$, we get

$$\log|S'_n| \approx \log|a| + \log|b| - bn,$$

which is a linear function of $n$. Therefore, by performing a linear fit to $\log|S'_n|$(for n indexing, say, the 30 most recent iterations), we can estimate $a$ and $b$, and then use these to estimate the relative error:

$$E_n = \frac{|S_\infty - S_n|}{S_\infty} \approx \frac{ae^{-bn}}{S_n}.$$

In this study we used finite differences to estimate the absolute derivative of $S_n$.

## Pre-iteration on coarser grids (PICG)

To further reduce the total simulation time, a pre-iteration on coarser grids (PICG) approach was applied (which is similar to the multilevel method [35]). Instead of running simulation directly on the original grid, we first perform simulations on a sequence of coarser grids, obtained by downsampling from the original (fine) grid, as illustrated in Figure 2. Starting from the coarsest grid, displacements obtained on each grid (for, say grid #2) were utilized as the initial displacements to CG iteration on the next grid (grid #1 in this case). This method significantly speeds up simulation since solutions obtained on coarser grids progressively approach the final solution. An overview of the entire problem-solving pipeline is provided in Figure 3, including image pre-processing, application of boundary conditions, pre-iteration on course grids, and parallelized conjugate gradient iteration.

## Experiments

To estimate actual memory usage and computation time as a function of the number of elements, fourteen sub-volumes of various sizes were extracted from a single 3D μCT image of a cadaver specimen of the human distal tibia (25 μm isotropic voxel size) and processed via simulated compression tests as described above. The simulation sizes for each sub-volume ranged between 1 and 75 million elements, corresponding to a range of 4.6 to 290 million variables for the linear systems. These were processed using 1, 2, 4, and 8 threads of execution.

In-vivo MR image data of the distal tibia of a postmenopausal woman previously acquired with a 3D fast spin-echo sequence [36] at $137\times137\times410\mu m^3$ resolution as part of an ongoing study to evaluate the effect of treatment with antiresorptive drugs were subjected to μFE analysis as described previously. The patient had been treated with zoledronic acid (Reclast$^{\text{TM}}$) and was examined at the start of intervention(baseline) and re-examined 12 months thereafter. Mechanical analysis was performed on both data sets (after mutual registration [37]) to evaluate the potential of the method to detect a possible improvement in the bone's mechanical

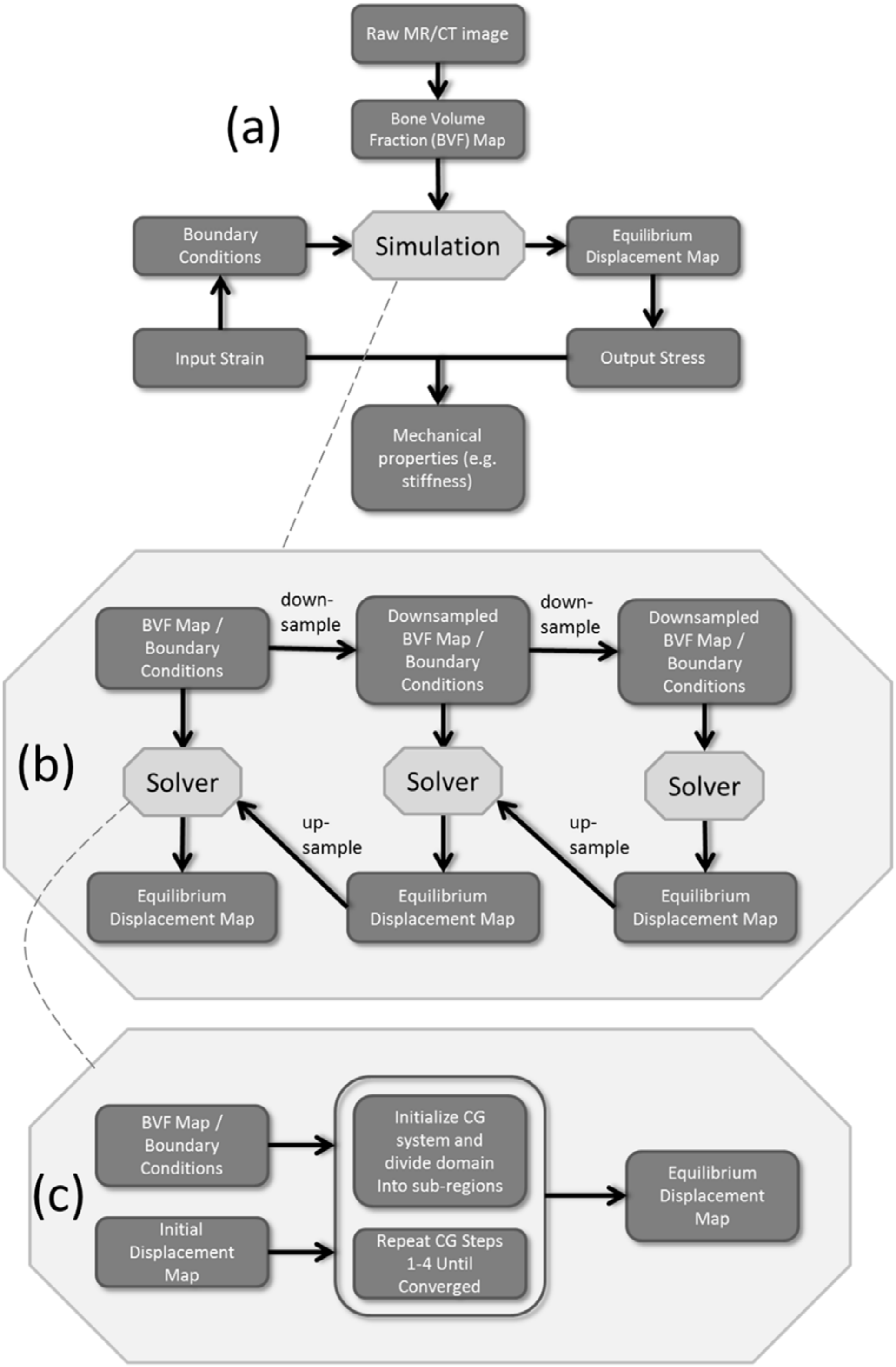
Raw MR/CT image
Bone Volume Fraction (BVF) Map
(a)
Boundary Conditions
Simulation
Equilibrium Displacement Map
Input Strain
Output Stress
Mechanical properties (e.g. stiffness)
(b)
BVF Map / Boundary Conditions
down-sample
Downsampled BVF Map / Boundary Conditions
down-sample
Downsampled BVF Map / Boundary Conditions
Solver
Solver
Solver
up-sample
up-sample
Equilibrium Displacement Map
Equilibrium Displacement Map
Equilibrium Displacement Map
(c)
BVF Map / Boundary Conditions
Initial Displacement Map
Initialize CG system and divide domain Into sub-regions
Repeat CG Steps 1-4 Until Converged
Equilibrium Displacement Map

**Figure 3. Flow diagram for the processing pipeline.** (a) Mechanical properties for the image-based finite-element model are obtained via the relationship between simulated input strain (applied boundary conditions) and resulting simulated stress, computed from the equilibrium displacement map. (b) The equilibrium displacement map is obtained using a series of parallelized conjugate gradient solvers (c) applied at a series of resolutions (pre-iteration on courser grids).
doi:10.1371/journal.pone.0035525.g003

competence in response to drug intervention. Structural parameters known to affect bone strength (e.g. bone volume fraction (BVF) and trabecular thickness) were evaluated at the two time-points as well.

In an additional experiment designed to evaluate the performance of the algorithm for processing of very large data sets, the proximal end of an intact human proximal femur was studied. The specimen was imaged by μCT on an X5000 industrial X-ray

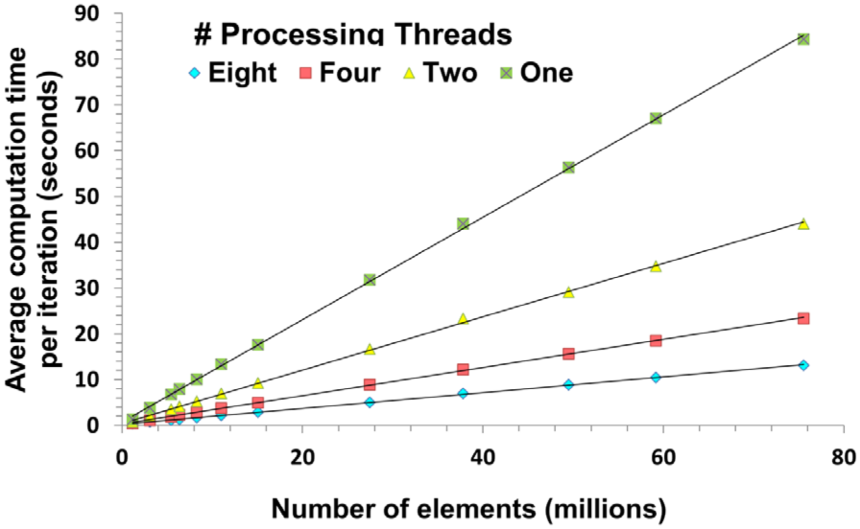

**Figure 4. Memory usage (GB) versus number of elements (millions) for μFE simulations on sub-volumes of a single μCT image using a single thread and eight threads.** The total number of elements in the μFE models ranged from 1 to 75 million. Symbols represent experimental data points, straight lines are best fits.
doi:10.1371/journal.pone.0035525.g004

inspection system (North Star Imaging, Rogers, MN) at an isotropic voxel size of 45 μm. The reconstructed images were then digitally stitched together to produce a single 3D volumetric image of matrix size 2444×1115×1770. The μFE solver was utilized to simulate compression applied to the top surface of the femoral head through a fictitious cap encompassing the top region of the femoral head (to mimic force transmitted through the acetabulum).

All simulations were performed on a laboratory desktop computer (dual quad core Xeon 3.16 GHz CPUs equipped with 40 GB of RAM).

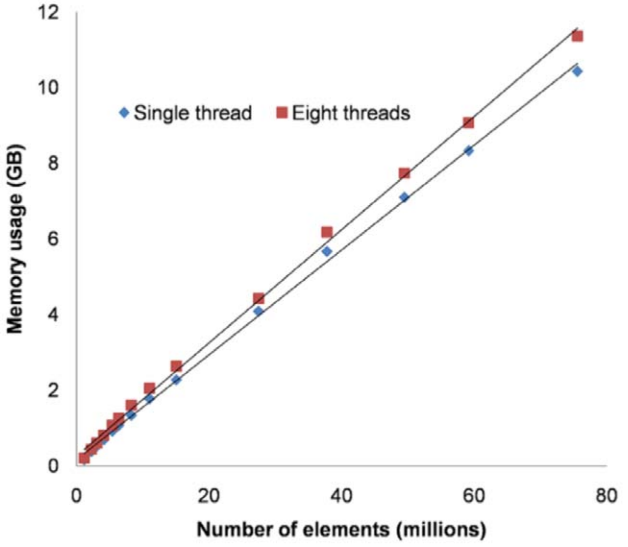

**Figure 5. Average time per iteration (seconds) versus number of elements (millions) for running μFE simulations on sub-volumes of a single μCT image using one, two, four, and eight threads respectively.** Symbols represent experimental data points, straight lines are best fits. Linear relationships were found in all cases with $R^2 \geq 0.998$.
doi:10.1371/journal.pone.0035525.g005

## Results

### Memory usage

Figure 4 provides the results of the memory usage experiments, showing the actual memory allocation during the conjugate gradient iterations as a linear function of the number of elements. The slopes suggest that 138 bytes per element are required for a single thread and 149 bytes per element when using eight threads. This is somewhat higher than the theoretical expectation of 130 bytes per element (single thread), partly because when estimating the theoretical expectation, approximations were made between number of elements, number of variables and the product of dimensions. These data suggest that on a system with 4 GB of RAM, we can expect to be able to simulate a system with ~20 million elements, whereas 100 million elements would be possible on a computer with 16 GB of RAM.

**Table 2.** Total number of iterations to reach 1% accuracy estimated with and without using the PICG approach in μFE simulations on sub-volumes with different number of elements.

| | Without PICG | With PICG ('4 2 1') |
|---|---|---|
| One-million-element data set | 224 | 88 |
| Three-million-element data set | 448 | 232 |
| Six-million-element data set | 459 | 240 |
| Eight-million-element data set | 476 | 235 |
| Eleven-million-element data set | 475 | 344 |
| Twenty-seven-million-element data set | 594 | 333 |
| Forty-nine-million-element data set | 598 | 317 |
| Seventy-five-million-element data set | 603 | 295 |

doi:10.1371/journal.pone.0035525.t002

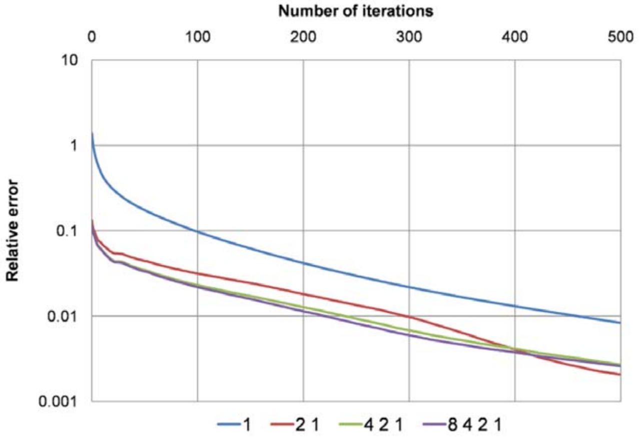

**Figure 6. Plots of the ''true'' relative errors on the finest grid obtained after using four different combinations of pre-iteration on coarser grids: '1', '2 1', '4 2 1', and '8 4 2 1'.** E.g., '4 2 1' means running simulations on data sets downsampled from the original data set by a factor of 4 and 2 sequentially, and then running simulations on the original data set. The combination '4 2 1' achieves the same accuracy as the more time-consuming combination, '8 4 2 1'.
doi:10.1371/journal.pone.0035525.g006

## Computation time

Figure 5 displays plots of average computation time per iteration versus number of elements for running μFE simulations on sub-volumes of a single μCT image using different numbers of threads: one, two, four, and eight. A speedup factor of 6.5 was observed when comparing eight processor threads to a single thread, compared with an ideal speedup factor of 8. The discrepancy is caused partly by interface communications between threads as well as the ~1–2% non-parallel part within each iteration, as described in the Methods. If 600 iterations are required based on the analysis above, then 2.2 hours would be needed to run a simulation on a data set with seventy-five million elements (using 8 threads). Furthermore, if applying the pre-iteration on coarser grids (PICG) approach, the total number of iterations on the finest grid could be reduced to 295 (see Table 2). Thus, approximately 1.3 hours (including computation time spent on the coarser grids) would be needed to run the same simulation in this context.

In Figure 6, the true relative errors (on the finest grid) obtained from using six different combinations of coarser grids in the PICG approach are plotted. For simulations on the coarser grids 200 iterations were used, whereas 500 iterations were used in simulation on the finest grid (the objective was to determine the true converged value, and then to retrospectively study the rate of convergence). As can be seen in Figure 6 the combination '4 2 1' achieves the same accuracy as the more time-consuming combination '8 4 2 1'.

Figure 7 provides a comparison of the true relative errors (on the finest grid) obtained using a different number of iterations on the downsampled grids using the combination '4 2 1', which was shown in Figure 6 to be optimal. To achieve 1% accuracy, around 200 iterations were needed on the finest grid when running 200 iterations on each of the coarser grids, while almost 400 iterations were needed on the finest grid when running 12 iterations on those coarser grids. Therefore the combination '4 2 1' was used in all subsequent experiments, with 200 iterations on the coarser grids.

In addition to the experimental results in terms of computation time using the authors' algorithms and programs, comparisons in computational performance with data reported in the literature are summarized in Table 3.

## Convergence criterion

Figures 8a and b show comparisons of different convergence criteria (in log scale) using two experiments (compression simulations on sub-volumes of the μCT image with 1 and 3 million elements respectively). The estimated relative error was obtained from applying the log-derivative approach as described in the Method Section; the scaled residual was obtained from scaling the ratio between the $L_2$ norm of the residual from each iteration and the $L_2$ norm of the right hand side of Eq. (6), which is equivalent to the ratio between the residual in the net force from each iteration and the force imposed on the boundary surfaces;

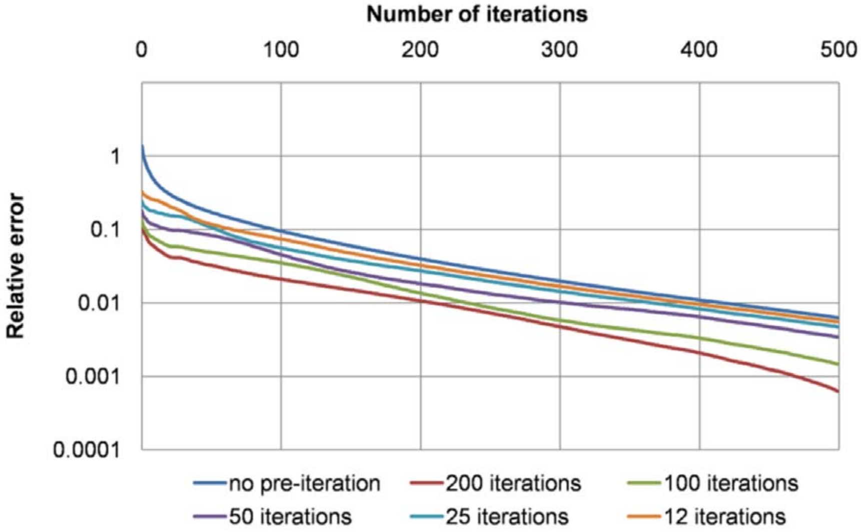

**Figure 7. Comparisons of the true relative errors on the finest grid obtained from running different number of iterations (200: red; 100: green; 50: purple; 25: blue; 12: orange) on all coarser grids in the combination strategy '4 2 1'.** Using 200 iterations on each coarser grid reduced the total number of iterations on the finest grid to around 200 compared to around 400 when using no pre-iteration for a 1% accuracy.
doi:10.1371/journal.pone.0035525.g007

**Table 3.** Comparisons in computational performance with literature-reported data.

| Ref. | Year | Anatomic Location | Number of Elements (million) | Number of Processors | FE Solver | Computer Type | Comp. Time/Million Elements |
|---|---|---|---|---|---|---|---|
| [29] | 1995 | N/A | 0.35 | not reported | N/A | Supercomputer | 12 h |
| [43] | 2003 | Femur | 96 | 30 | N/A | SGI-Origin2000 | 260.4cpuh |
| [43] | 2003 | Femur | 33 | 16 | N/A | SGI-Origin3800 | 606.1cpuh |
| [40] | 2004 | N/A | 135 | 4000 | Olympus | IBM SP Power3 | ~25.2 s |
| [44] | 2006 | N/A | ≤2.7 | not reported | Olympus | Cray-Dell PowerEdge Xeon cluster parallel supercomputer | 4.5 h** |
| [45] | 2007 | N/A | 5.44 | 256 | ParFE | Cray XT3 | 10.8 s |
| [46] | 2008 | N/A | 247.73 | 1024 | ParFE | Cray XT3 | 2.9 s |
| [47] | 2009 | radius & tibia | 8 | not reported | Scanco | Workstation | 45 m |
| [30] | 2010 | N/A | ~375* | 8192 | ParFE | IBM Blue Gene | ~3.6 s* |
| [39] | 2011 | Radius | 2 | not reported | Scanco | N/A | 1.5 h |
| [39] | 2011 | Tibia | 5 | not reported | Scanco | N/A | 1 h |
| | **2011** | **Tibia** | **0.2** | **8** | **FESBI** | **desktop computer** | **4.52 m** |
| | **2011** | **Femur** | **90.3** | **8** | **FESBI** | **desktop computer** | **2.25 m** |

Note: numbers marked with.
*are approximations based on reported number of degrees of freedom; numbers with.
**are approximated averages; **the last two rows (bold) are performance results from the authors' software**. Although difficult to compare different methodologies (with varying parameters, etc), these numbers are accurate to the best of the authors' knowledge.

doi:10.1371/journal.pone.0035525.t003

and the true relative error was obtained using $\frac{\|x-x_{true}\|}{\|x_{true}\|}$, where $x$ is the computed solution (the primary stress in our case) from each iteration and $x_{true}$ is the true solution obtained from running many more iterations than needed.

Retrospectively tested, the estimated relative error has similar trends as the true relative error while the scaled residual does not. Furthermore, the estimated relative error is accurate to within a factor of at most two of the true relative error and tends to be increasingly accurate as the iteration number increases.

Table 2 lists the total numbers of iterations required to achieve around 1% accuracy of the true solution (the "infinitely converged" solution) for data sets with different numbers of elements. Results with and without applying the PICG approach are given, showing that using PICG reduces the total number of iterations required to reach a 1% accuracy by a factor of ~2. Based on our experimental results (see Figures 6 and 7), the combination strategy '4 2 1' was utilized here in the PICG approach.

### Applications to trabecular bone mechanics

Simulated axial compression tests of the 7T MRI patient image data for the two 3D data sets described in the Methods section (baseline and 12-month follow-up) contained approximately 0.2 million elements taking 27 seconds to converge. Measured bone-volume fraction, trabecular thickness and axial stiffness were found to have increased over the course of the one-year treatment period. BVF had increased from 7.4 to 8.7%; trabecular thickness from 99.5 to 102.4 μm; and axial stiffness from 247 to 293 MPa. Figure 9 displays mutually registered parametric strain energy maps at the two time-points. The images at the two time-points show remarkable similarity indicative of relatively small remodeling changes (Figures 9b and f) which, however, appear to have significant mechanical consequences as suggested by the 19% increase in predicted stiffness. We also notice a rather unequal loading pattern exhibiting greater strain medially than laterally at least in the anterior region displayed in Figures 9d and f.

In an additional experiment designed to evaluate the performance of the algorithm for processing of very large data sets, the proximal end of an intact human proximal femur was studied. A sample slice of the simulated strain-energy map in coronal view is given in Figure 10. The total number of elements in the μFE model for the downsampled data was 90.3 million and the total time for solving the resultant linear system was 6.8 hours on the desktop computer (as described in the Methods) using parallel computing with eight threads. The PICG approach was also applied for comparison. Because of the large size of the femur data, a combination '32 16 8 4 2 1' was utilized where a total of 200 iterations were performed on simulations based on the coarse grids, thereby reducing total computation time to 2.3 hours.

## Discussion

We have conceived and implemented substantially improved algorithms yielding a computationally efficient program for large-scale finite-element simulations of bone mechanics on the basis of μMR and μCT images. The work's primary goal was to enable the performance of such simulations on desktop computers within practical computation times. Three key factors for improving efficiency were investigated: memory usage, computation time and convergence criteria.

The implementation of the EBE approach [32,33] avoids storing the entire linear system thereby greatly reducing memory usage. The theoretical memory savings (from using EBE), was estimated in this work as a factor of 10, agreeing well with the estimate of a factor of 9 from [38]. A row-by-row (RBR) approach was also proposed in [38], but based on numerical examples, the RBR approach appears to use as much as 35% more memory than the EBE approach.

Thus far, computational constraints on desktop computers limited image-based FE analyses of bone structure networks to

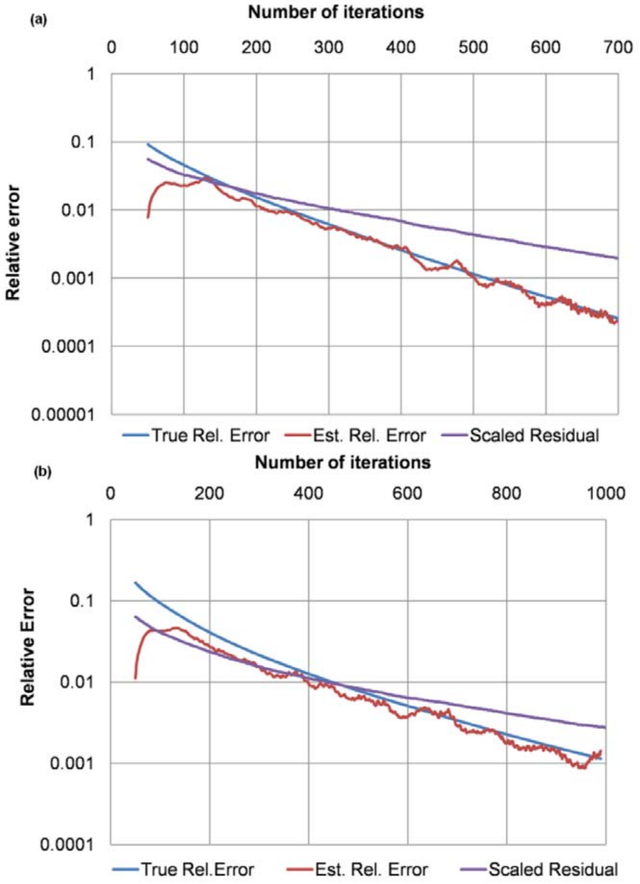

**Figure 8. Convergence criteria comparisons.** (a) in example 1, the estimated relative error (red) has almost the same trend as the true relative error (blue), whereas the scaled residual (purple) deviates substantially; (b) the estimated relative error (red) in example 2 also better approximates to the true relative error (blue) than the scaled residual (purple).
doi:10.1371/journal.pone.0035525.g008

models with relatively small numbers of finite elements. Simulation of a distal tibia model with 5 million elements, for example, typically takes around 5 hours as reported in recent work [39]. With the enhanced FE algorithms detailed in the present work, a bone structure model of this complexity can be solved in 9 minutes to achieve 1% estimated accuracy in the output parameter.

Large-scale FE simulations of trabecular bone networks have previously been performed on highly scalable, implicitly parallel, one-of-a-kind supercomputers. For example, Adams et al. in 2004 solved a FE model of a vertebral body with 135 million elements using an ASCI White supercomputer consisting of 292 computer nodes [40]. A recent study reported solving FE systems with up to about 1.5 billion unknowns (~375 million elements) within half an hour using 8192 cores of the Blue Gene/L supercomputer at IBM T.J. [30]. Our current desktop FE implementation can achieve a comparable task consisting of a system with 135 million elements in approximately 6 hours utilizing 20 GB of memory. Currently, with our 8-core laboratory computer is equipped with 40 GB of memory, we project a maximum solvable system size of around 266 million elements in ~13 hours. To the best of the authors' knowledge, solving of such large-scale models of trabecular bone networks on a single desktop computer has not previously been feasible.

We compared the computation time achievable by the present work against literature-reported values (see Table 3). It is to be noted that the majority of the work reported therein has been performed on large-scale computer clusters, with hundreds to thousands of processors using canned software packages. While one to two orders slower than such computers, our system far outperforms desktop-based systems currently in use. Furthermore, the present program has been designed from scratch to optimally handle bone structural images in terms of its I/O capabilities and its computational efficiency is achieved with a mere eight cores of a standard, readily available, desktop computer. With simulation times on the order of minutes for typical array sizes for in vivo images on the order of 1–2 million elements, on-line computation as part of the image reconstruction and processing pipeline now has become practical. The system makes simulations on very large arrays such as those resulting from whole-bone μCT images feasible, which previously required access to supercomputers.

Parallel programming on desktop computers is becoming increasingly attractive with the availability of multi-core computers systems. With the trend of continually increasing the number of cores on a single computer (e.g. 16 or 32 cores), even larger FE systems than those demonstrated here can be expected to be solved on desktop systems in the near future without the need for supercomputers with the algorithmic optimizations described in this work. In some situations, computation time could be further reduced by adapting the present methodology to graphics processing unit (GPU) computing (substantially increasing the parallelization factor). However, at present, the available RAM on a GPU is often limited to around 1 GB, precluding simulations involving more than around 10 million elements.

Comparison of computation times reported by different studies is often not straightforward because convergence criteria are not explicitly stated. The magnitude of the residual at a given iteration is widely used to decide when to stop the simulation. However, since the residual is computed as an internal step while solving the linear system of equations, it is not directly reflective of the magnitude of error in the output parameter (stiffness, for example) at a given FE iteration. To overcome these limitations, a novel convergence criterion was adopted in our FE implementation, which indicates how close the computed stiffness value at each iteration is to the "true" value. For this study, a 1% error in stiffness was used as a convergence criterion. Our experiments suggest that the total number of iterations needed using the new convergence criterion is closer to the actual requirement than that estimated using other convergence criteria, for instance, estimations based on (relative/scaled) residual. With the PICG approach the total number of iterations has been shown to decrease from 485±125 (without PICG) to 260±83.

Compared to general-purpose FE software, the computational infrastructure described here provides a number of advantages for the target application of high-resolution image-based bone biomechanics. First, an integrated interface is provided to directly import raw medical images (including k-space) data for analysis, thereby eliminating the need for additional file conversion software. Second, since the FE model is generated by one-to-one mapping of image voxels into finite elements, the mechanical estimates are not influenced by differences in various mesh-generation methods, which are also computationally demanding for large systems [41]. Third, the program can operate in the binary as well as gray-scale mode, customized for generating FE models on the basis of high-resolution (e.g. micro-CT) and in-vivo (e.g. MR or CT) images of bone to account for partial volume mixing, or regional variations in attenuation coefficients.

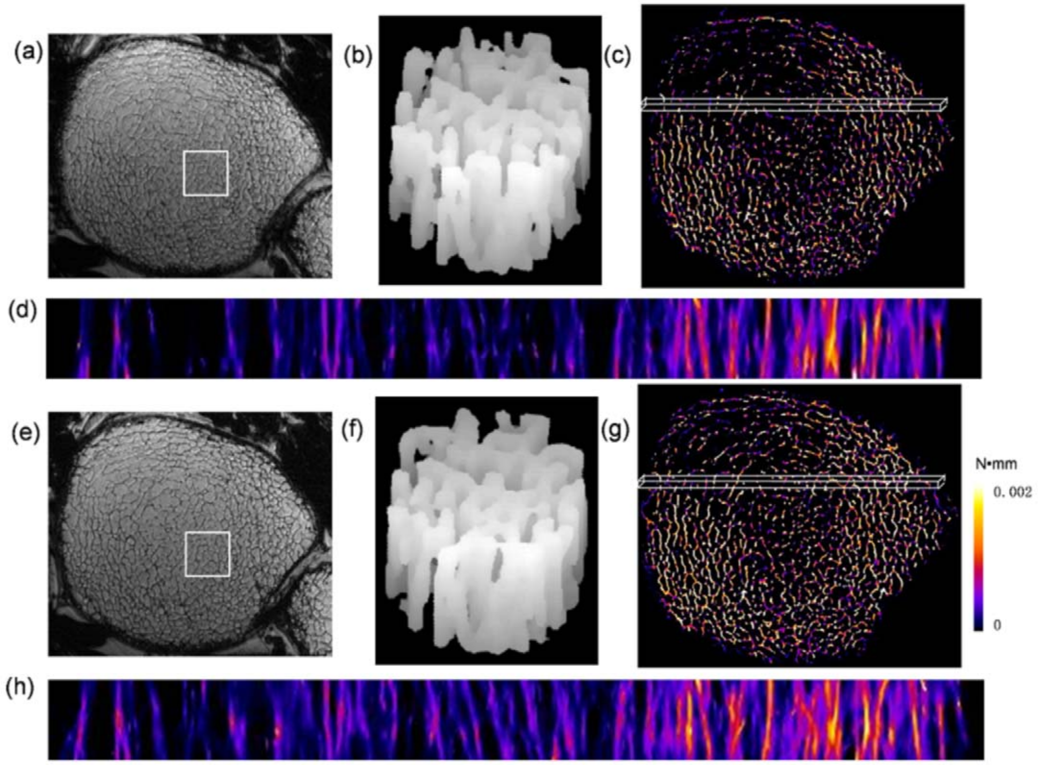

**Figure 9. Structural MR images and resulting strain energy maps from two 3D data sets taken at 7T field strength at the left distal femur in a patient treated with zoledronic acid (Reclast$^{TM}$) for a period of 12 months.** a, e) cross-sectional images from stack of contiguous slices at baseline and after 12 months of treatment; b, f) volume-renderings of square-shaped sub-regions (7.8×7.8×3.7 mm$^3$) indicated in images a and e, highlighting structural similarity at the two time-points; d, h) longitudinal projections of strain-energy data displayed for a thin slab (1.1×37.7×3.7 mm$^3$) in the anterior region indicated in c and g.
doi:10.1371/journal.pone.0035525.g009

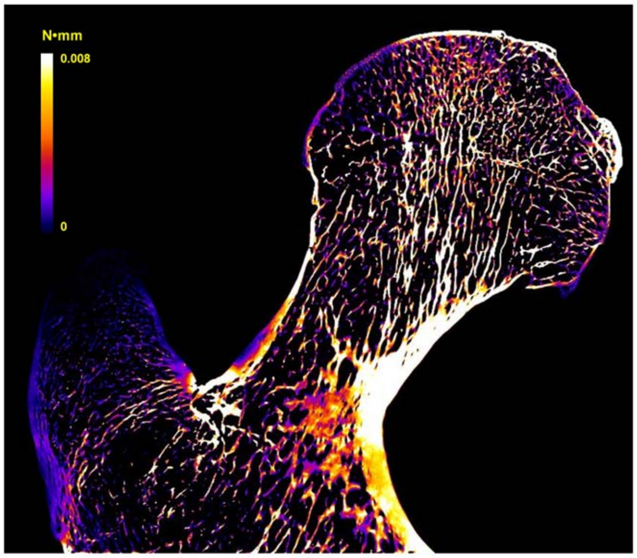

**Figure 10. Sample slice of simulated strain-energy map of a human femur in coronal view.** To simulate a compression along the z-axis, a 1% strain was applied to the top surface of the femoral head. For details see text.
doi:10.1371/journal.pone.0035525.g010

Some limitations of the present work are noted. So far, we have confined the analysis presented here to the linear elastic regime, although nonlinear FE modeling can provide additional information on bone's failure mechanisms [42]. We expect that the substantial improvements in speed and resource utilization achieved under the present work will make nonlinear analysis feasible within clinically acceptable computation time limits. Nonlinear analysis typically entails application of a series of incremental strains with each step involving solution of a linear system. Therefore, the present methodology may substantially improve the efficiency of non-linear analyses. We project that nonlinear analysis on an in-vivo MRI data set on the order of one million elements could be tackled within an hour or at least in the time needed currently with commercial desktop based systems for linear analysis.

In conclusion, the desktop computer based FE approach detailed here enables computational biomechanics of bone, previously confined to research studies, in clinical settings.

## Supporting Information

**Appendix S1 Derivation of the kernel matrix for a single element.**
(DOC)

## Acknowledgments

The authors would like to thank Dr. Alexander Wright and Dr. Yusuf Bhagat of the Laboratory for Structural NMR Imaging for providing experimental data, and Dr. Michael Wald for providing software to do volume rendering.

## Author Contributions

Conceived and designed the experiments: JM NZ CR FW. Performed the experiments: JM NZ. Analyzed the data: JM NZ. Contributed reagents/materials/analysis tools: JM. Wrote the paper: JM NZ CR FW.

## References

1. Mcculloch A, Waldman L, Rogers J, Guccione J (1992) Large-Scale Finite-Element Analysis of the Beating Heart. Critical Reviews in Biomedical Engineering 20: 427–449.
2. Hou TY, Wu XH (1997) A multiscale finite element method for elliptic problems in composite materials and porous media. Journal of Computational Physics 134: 169–189.
3. Van Rietbergen B, Muller R, Ulrich D, Ruegsegger P, Huiskes R (1999) Tissue stresses and strain in trabeculae of a canine proximal femur can be quantified from computer reconstructions. J Biomech 32: 443–451.
4. Danilov S, Kivman G, Schroter J (2004) A finite-element ocean model: principles and evaluation. Ocean Modelling 6: 125–150.
5. Zhang XH, Liu XS, Vasilic B, Wehrli FW, Benito M, et al. (2008) In vivo microMRI-based finite element and morphological analyses of tibial trabecular bone in eugonadal and hypogonadal men before and after testosterone treatment. J Bone Miner Res 23: 1426–1434.
6. Wehrli FW, Rajapakse CS, Magland JF, Snyder PJ (2010) Mechanical implications of estrogen supplementation in early postmenopausal women. J Bone Miner Res 25: 1406–1414.
7. Silva MJ, Keaveny TM, Hayes WC (1998) Computed tomography-based finite element analysis predicts failure loads and fracture patterns for vertebral sections. J Orthop Res 16: 300–308.
8. Vilayphiou N, Boutroy S, Sornay-rendu E, Van rietbergen B, Munoz F, et al. (2010) Finite element analysis performed on radius and tibia HR-pQCT images and fragility fractures at all sites in postmenopausal women. Bone 46: 1030–1037.
9. Podshivalov L, Fischer A, Bar-Yoseph PZ (2011) 3D hierarchical geometric modeling and multiscale FE analysis as a base for individualized medical diagnosis of bone structure. Bone 48: 693–703.
10. Jaworski ZFG (1987) Physiology and pathology of bone remodeling. Cellular basis of bone structure in health and in osteoporosis. Orthopedic Clinics of North America 12: 485.
11. Recker R, Lappe J, Davies KM, Heaney R (2004) Bone remodeling increases substantially in the years after menopause and remains increased in older osteoporosis patients. J Bone Miner Res 19: 1628–1633.
12. Muller R (2004) Long-term prediction of three-dimensional bone architecture in simulations of pre-, peri- and post-menopausal microstructural bone remodeling. Osteoporos Int.
13. Vico L, Chappard D, Palle S, Bakulin AV, Novikov VE, et al. (1988) Trabecular bone remodeling after seven days of weightlessness exposure (BIOCOSMOS 1667). Am J Physiol 255: R243–247.
14. Saha PK, Wald MJ, Radin A, Wehrli FW (2005) Predicting mechanical competence of trabecular bone using 3D tensor-scale-based parameters. Proc SPIE 5746: 12.
15. Wehrli FW (2007) Structural and functional assessment of trabecular and cortical bone by micro magnetic resonance imaging. J Magn Reson Imaging 25: 390–409.
16. Zikan V, Tyblova M, Raska I, Michalska D, Luchavova M (2011) Bone Health in Patients with Multiple Sclerosis Treated with Low Dose Glucocorticoids. Osteoporosis International 22: 391–391.
17. Ulrich D, Hildebrand T, Van Rietbergen B, Muller R, Ruegsegger P (1997) The quality of trabecular bone evaluated with micro-computed tomography, FEA and mechanical testing. Stud Health Technol Inform 40: 97–112.
18. Ladd AJ, Kinney JH, Haupt DL, Goldstein SA (1998) Finite-element modeling of trabecular bone: comparison with mechanical testing and determination of tissue modulus. J Orthop Res 16: 622–628.
19. Kabel J, van Rietbergen B, Dalstra M, Odgaard A, Huiskes R (1999) The role of an effective isotropic tissue modulus in the elastic properties of cancellous bone. J Biomech 32: 673–680.
20. Boutroy S, Van Rietbergen B, Sornay-Rendu E, Munoz F, Bouxsein ML, et al. (2008) Finite element analysis based on in vivo HR-pQCT images of the distal radius is associated with wrist fracture in postmenopausal women. J Bone Miner Res 23: 392–399.
21. Wehrli FW, Hwang SN, Ma J, Song HK, Ford JC, et al. (1998) Cancellous bone volume and structure in the forearm: noninvasive assessment with MR microimaging and image processing [published erratum appears in Radiology 1998 Jun;207(3):833]. Radiology 206: 347–357.
22. Boutry N, Cortet B, Dubois P, Marchandise X, Cotten A (2003) Trabecular bone structure of the calcaneus: preliminary in vivo MR imaging assessment in men with osteoporosis. Radiology 227: 708–717.
23. Hwang SN, Wehrli FW (2002) Subvoxel Processing: A Method for reducing partial volume blurring with application to in vivo MR images of trabecular bone. Magnetic Resonance in Medicine 47: 948–957.
24. Magland JF, Wehrli FW (2008) Trabecular bone structure analysis in the limited spatial resolution regime of in vivo MRI. Acad Radiol 15: 1482–1493.
25. Rajapakse CS, Magland JF, Wald MJ, Liu XS, Zhang XH, et al. (2010) Computational biomechanics of the distal tibia from high-resolution MR and micro-CT images. Bone 47: 556–563.
26. Macneil JA, Boyd SK (2008) Bone strength at the distal radius can be estimated from high-resolution peripheral quantitative computed tomography and the finite element method. Bone 42: 1203–1213.
27. Dempster WT, Liddicoat RT (1952) Compact bone as a non-isotropic material. American Journal of Anatomy 91: 32.
28. Goulet RW, Goldstein SA, Ciarelli MJ, Kuhn JL, Brown MB, et al. (1994) The relationship between the structural and orthogonal compressive properties of trabecular bone. Journal of Biomechanics 27: 375–389.
29. van Rietbergen B, Weinans H, Huiskes R, Odgaard A (1995) A new method to determine trabecular bone elastic properties and loading using micromechanical finite-element models. Journal of Biomechanics 28: 69–81.
30. Bekas C, Curioni A, Arbenz P, Flaig C, Van Lenthe GH, et al. (2010) Extreme scalability challenges in micro finite element simulations of human bone. Concurrency and Computation: Practice and Experience 22: 2282–2296.
31. Saad Y, Saad Y (1996) Iterative methods for sparse linear systems: PWS Pub. Co.
32. Hughes TJR, Levit I, Winget J (1983) An Element-by-Element Solution Algorithm for Problems of Structural and Solid Mechanics. Computer Methods in Applied Mechanics and Engineering 36: 241–254.
33. Hughes TJR, Ferencz RM, Hallquist JO (1987) Large-Scale Vectorized Implicit Calculations in Solid Mechanics on a Cray X-Mp/48 Utilizing Ebe Preconditioned Conjugate Gradients. Computer Methods in Applied Mechanics and Engineering 61: 215–248.
34. Hendrickson B, Leland R, Plimpton S (1993) An Efficient Parallel Algorithm for Matrix-Vector Multiplication. Sandia Report SAND92-2765, Sandia National Lab, Albuquerque, NM 87185.
35. Teng SH (1999) Coarsening, sampling, and smoothing: Elements of the multilevel method. Algorithms for Parallel Processing 105: 247–276.
36. Magland JF, Rajapakse CS, Wright AC, Acciavatti R, Wehrli FW (2010) 3D fast spin echo with out-of-slab cancellation: a technique for high-resolution structural imaging of trabecular bone at 7 Tesla. Magn Reson Med 63: 719–727.
37. Magland JF, Jones CE, Leonard MB, Wehrli FW (2009) Retrospective 3D registration of trabecular bone MR images for longitudinal studies. J Magn Reson Imaging 29: 118–126.
38. VanRietbergen B, Weinans H, Huiskes R, Polman BJW (1996) Computational strategies for iterative solutions of large FEM applications employing voxel data. International Journal for Numerical Methods in Engineering 39: 2743–2767.
39. Vilayphiou N, Boutroy S, Szulc P, Van Rietbergen B, Munoz F, et al. (2011) Finite element analysis performed on radius and tibia HR pQCT images and fragility fractures at all sites in men. Journal of Bone and Mineral Research 26: 965–973.
40. Adams MF, Bayraktar HH, Keaveny TM, Papadopoulos P (2004) Ultrascalable implicit finite element analyses in solid mechanics with over a half a billion degrees of freedom. ACM/IEEE Proceedings of SC2004: High Performance Networking and Computing. 34 p.
41. Bern M, Eppstein D (1992) Mesh generation and optimal triangulation. Computing in Euclidean geometry 1: 23–90.
42. Bevill G, Keaveny TM (2009) Trabecular bone strength predictions using finite element analysis of micro-scale images at limited spatial resolution. Bone 44: 579–584.
43. Zienkiewicz OC, Taylor RL, Zhu JZ The Finite Element Method: Its Basis and Fundamentals, Sixth Edition: Butterworth-Heinemann.
44. Van Rietbergen B, Huiskes R, Eckstein F, Ruegsegger P (2003) Trabecular bone tissue strains in the healthy and osteoporotic human femur. J Bone Miner Res 18: 1781–1788.
45. Bevill G, Eswaran SK, Gupta A, Papadopoulos P, Keaveny TM (2006) Influence of bone volume fraction and architecture on computed large-deformation failure mechanisms in human trabecular bone. Bone 39: 1218–1225.
46. Arbenz P, van Lenthe G, Mennel U, Muller R, Sala M (2007) Multi-level μ-finite element analysis for human bone structures. Applied Parallel Computing State of the Art in Scientific Computing. pp 240–250.
47. Arbenz P, Van Lenthe GH, Mennel U, Muller R, Sala M (2008) A Scalable Multi-level Preconditioner for Matrix-Free μ-Finite Element Analysis of Human Bone Structures. Numerical Methods in Engineering. 21 p.
48. Dalzell N, Kaptoge S, Morris N, Berthier A, Koller B, et al. (2009) Bone micro-architecture and determinants of strength in the radius and tibia: age-related changes in a population-based study of normal adults measured with high-resolution pQCT. Osteoporosis international 20: 1683–1694.