DCEMRI.jl: A fast, validated, open source toolkit for dynamic contrast enhanced MRI analysis

David S Smith, Xia Li, Lori Arlinghaus, Thomas E Yankeelov, E Brian Welch

We present a fast, validated, open-source toolkit for processing dynamic contrast enhanced magnetic resonance imaging (DCE-MRI) data. We validate it against the Quantitative Imaging Biomarkers Alliance (QIBA) Standard and Extended Tofts-Kety phantoms and find near perfect recovery in the absence of noise, with an estimated 10-20x speedup in run time compared to existing tools. To explain the observed trends in the fitting errors, we present an argument about the conditioning of the Jacobian in the limit of small and large parameter values. We also demonstrate its use on an in vivo data set to measure performance on a realistic application. For a 192 x 192 breast image, we achieved run times of < 1 s. Finally, we analyze run times scaling with problem size and find that the run time per voxel scales as O($N^{1.9}$), where N is the number of time points in the tissue concentration curve. DCEMRI.jl was much faster than any other analysis package tested and produced comparable accuracy, even in the presence of noise.

DCEMRI.jl: A Fast, Validated, Open Source Toolkit for Dynamic Contrast Enhanced MRI Analysis

David S. Smith[1,2]**, Xia Li**[1,2]**, Lori R. Arlinghaus**[1,2]**, Thomas E. Yankeelov**[1,2,4,5]**, and E. Brian Welch**[1,2,3]

[1]**Institute of Imaging Science, Vanderbilt University, Nashville, Tennessee, USA 37232**
[2]**Department of Radiology and Radiological Sciences, Vanderbilt University**
[3]**Department of Biomedical Engineering, Vanderbilt University**
[4]**Department of Physics and Astronomy, Vanderbilt University**
[5]**Department of Cancer Biology, Vanderbilt University**

ABSTRACT

We present a fast, validated, open-source toolkit for processing dynamic contrast enhanced magnetic resonance imaging (DCE-MRI) data. We validate it against the Quantitative Imaging Biomarkers Alliance (QIBA) Standard and Extended Tofts-Kety phantoms and find near perfect recovery in the absence of noise, with an estimated 10–20$\times$ speedup in run time compared to existing tools. To explain the observed trends in the fitting errors, we present an argument about the conditioning of the Jacobian in the limit of small and large parameter values. We also demonstrate its use on an in vivo data set to measure performance on a realistic application. For a 192×192 breast image, we achieved run times of < 1 s. Finally, we analyze run times scaling with problem size and find that the run time per voxel scales as $O(N^{1.9})$, where N is the number of time points in the tissue concentration curve. `DCEMRI.jl` was much faster than any other analysis package tested and produced comparable accuracy, even in the presence of noise.

INTRODUCTION

Dynamic contrast enhanced MRI

Dynamic contrast enhanced magnetic resonance imaging (DCE-MRI) involves the continuous acquisition of heavily T_1-weighted MR images while a paramagnetic contrast agent (CA) is injected. The CA increases the contrast between different tissues by changing their inherent relaxation rates. By collecting serial images, each image voxel yields an intensity time course that can be used to estimate physiological parameters, such as the volume transfer constant K^{trans}, extravascular extracellular volume fraction v_e, and the plasma volume fraction v_p (Choyke et al., 2003; Yankeelov and Gore, 2009). Due to this, DCE-MRI has successfully been applied to assess vascular characteristics in both pre-clinical (Zwick et al., 2009; Jensen et al., 2010) and clinical settings (Lockhart et al., 2010; Mannelli et al., 2010))

The MR scanner typically handles the reconstruction of the acquired raw MR data into images, while the second step of DCE-MRI analysis is left to the end user, at least in research settings. This second step includes determining a subset of voxels to process, fitting a nonlinear signal model to the time curve of each of those voxels, postprocessing the fitted model parameters, and summarizing the results.

Existing Analysis Software

Several DCE-MRI analysis packages have been released to the community. *DCE@urLAB*[1] from Ortuño et al. (2013) has been validated against reference phantoms and includes many pefusion models, but it requires IDL, a commercial software package. It can be run for free with the IDL Virtual Machine, which

---

[1]`http://www2.die.upm.es/im/archives/DCEurLAB/`

requires registration and approval from the vendor. And in the end the full IDL development environment must be installed, even though its functionality is crippled without a paid license. Second, DCE@urLAB was built around a graphical user interface (GUI) that did not provide batch processing. It also did not run on Mac OS X or Linux. Finally, the stated run times (Ortuño et al., 2013) were slower than we expected, given the computational complexity of nonlinear least squares fitting, suggesting some inefficiency due to the software complexity (∼20,000 lines of IDL code) or GUI overhead.

Zöllner et al. (2013) built an OsiriX plug-in called *UMMPerfusion*.[2] This is a validated, open-source software package for DCE-MRI analysis, but it works only within the OsiriX image viewer, a Mac OS X only program and which only a 32-bit, basic version was available for free. This package also did not provide a headless batch processing tool, and the reported run times were also slower than expected, even though it supported parallel processing.

The DCE Tool[3] is another GUI solution for DCE modeling. It requires both a Windows only software package called ClearCanvas (available for free) and the MATLAB[4] run time environment (a commercial product) in order to run. This solution was quite complex, comprising over 1.2 million lines of C and C# code.

The most useful existing tool for our needs was `dcemriS4`[5] (Schmid et al., 2009a,b, 2006), an R package. R is a popular statistical analysis language, but it is not as common in MRI research. Nevertheless, the package contained many advanced models, was fast in our benchmarks, and included parallel processing, but it still took tens of seconds to process a typical breast DCE-MRI data set and to the best of our knowledge has not been validated.

Why Julia?

We desired a fast, free, simple, easily extendable code that required a minimal installation, would work on Windows, Mac OS X, and Linux, and would be familiar to MATLAB users. Based on these criteria, we chose Julia as the implementation language. Julia (Bezanson et al., 2012, 2014) is a new, high-level language designed for technical computing and that approaches the performance of C and Fortran.[6] It contains extensive libraries for linear algebra and signal processing and provides distributed parallel execution. It also easily interoperates with existing scientific languages, such as C, Fortran, and Python, making it an excellent glue language for scientific computing. Interested readers can get a flavor of Julia's simultaneous efficiency and simplicity with the example in Listing 1.

In short, Julia feels like MATLAB, which is simple and familiar to many investigators, but runs faster and is completely free. In particular for DCE-MRI, Julia's simple and flexible parallel computing model allows excellent parallelization of the nonlinear least squares fitting problem.

Goals

We developed `DCEMRI.jl` with five features in mind: open source, free, fast, flexible, and simple. Open source enables auditing, bug finding, and community improvement. Free software reduces the barrier to use and adoption. A faster package saves investigator time and is more clinically practical. Flexibility anticipates new uses and heterogeneous adoption. And simple code design reduces bugs through easier auditing and yields greater didactic value. Here we present the results of the effort to produce such a package.

MATERIALS AND METHODS

Units

All input variables are assumed to be in SI units, except flip angles which should be given in degrees. All output is in SI units except for K^{trans} which is scaled to units of min^{-1} to maintain convention within the DCE-MRI community.

---

[2]http://ikrsrv1.medma.uni-heidelberg.de/redmine/projects/ummperfusion
[3]http://thedcetool.com/
[4]Mathworks, Natick, MA
[5]http://dcemri.sourceforge.net/
[6]http://julialang.org/benchmarks/

```
julia> f(x,y) = x + y
f (generic function with 1 method)

julia> f(2,3)
5

julia> code_native(f, (Float64, Float64))
.section __TEXT,__text,regular,pure_instructions
push RBP
mov RBP, RSP
vaddsd XMM0, XMM0, XMM1
pop RBP
ret
```

Listing 1—The function $f(x,y)$ is compiled upon first execution using Julia's just-in-time (JIT) compiler, making subsequent function calls much faster. The generated CPU instructions can be viewed directly with the `code_native` function, which takes two arguments: a function name and an n-tuple of argument types. Here we examine the native CPU code produced if f were called on two 64-bit floating point variables. The native code contains only one math instruction (`vaddsd`), surrounded by the code required to preserve and restore the stack for a function call. (By convention the first two floating point arguments are passed into the function in the `XMM0` and `XMM1` registers, and the result is placed in `XMM0`.) All other overhead has been stripped away automatically without an explicit compilation. The development and testing cycle is accelerated by this elimination of a separate code compilation step, and the generated code executes as quickly as compiled C or Fortran would.

Reproducible Research

76

77 `DCEMRI.jl` is maintained under version control in a Github[7] archive. The exact version of the code
78 used to produce the results here may be obtained by "pinning" module at version 0.1.0 using the command
79 `Pkg.pin("DCEMRI", v"0.1.0")` in the Julia shell. This will effectively reverse any code changes
80 committed to the repository after the publication of this paper. The command `Pkg.free("DCEMRI")`
81 will "unpin" and get the latest updates again.

Input Parameters

82

83 For simplicity and maximum compatibility, `DCEMRI.jl` reads and writes input as Matlab MAT v5 files.
84 This allows users to call `DCEMRI.jl` from any language that can read and write MAT files. This list
85 includes MATLAB, Octave, Python, and R. The input MAT file must include a vector (`t`) of imaging time
86 points, the arterial input function (`Cp`), the DCE data (`DCEdata`), and either a map of the pre-contrast R_1
87 relaxation rate $R_1(\mathbf{x},0)$ and associated signal $S(\mathbf{x},0)$ (`R10`, `S0`) or a series of T_1-weighted multiflip data
88 (`T1data`) and associated flip angles (`T1flip`). All other parameters are optional and will be supplied
89 with defaults if not provided. The defaults may be overridden by supplying additional parameters in the
90 MAT file, command-line arguments, or function parameters.

T1 mapping

91

`DCEMRI.jl` can accept as input an R_1 longitudinal relaxation rate map ($R_1 \equiv 1/T_1$) or multiple flip angle (multiflip) T_1-weighted dynamic data. If the multiflip data is supplied, the code will fit R_1 and signal density maps using the signal equation for a spoiled gradient echo sequence:

$$S(\mathbf{x},t) = S(\mathbf{x},0)\sin\theta\frac{1-\exp[-R_1(\mathbf{x},t)T_R]}{1-\cos\theta\exp[-R_1(\mathbf{x},t)T_R]}, \tag{1}$$

---

[7]`http://github.com/davidssmith/DCEMRI.jl`

where $S(\mathbf{x},t)$ is the signal as a function of space $\mathbf{x}$ and time t, θ is the flip angle, and T_R is the repetition time. Here we have ignored R_2^* decay because we are assuming that the echo time T_E is much shorter than $1/R_2^*$.

If an R_1 fit is required, only voxels with a mean signal intensity of at least 10% of the maximum intensity are fit, to avoid fitting voxels that are dominated by noise. This cutoff was chosen empirically and can be customized to the signal distribution in a given data set. Voxels selected for fitting are then split evenly across CPU cores for Levenberg-Marquardt fitting to the signal equation. The results for each voxel subset are returned to the parent process, where full maps of $R_1(\mathbf{x},0)$ and $S(\mathbf{x},0)$ are formed. The fact that fitting of individual voxels is independent of neighboring voxels is crucial to allowing this problem to be efficiently parallelized. All models implemented so far in `DCEMRI.jl` take voxels to be independent of their neighbors.

DCE fitting

To fit a model to the supplied DCE data, the raw MR signal is converted first to an effective $R_1(\mathbf{x},t)$ relaxation rate by inverting the signal equation (Eq. 1):

$$R_1(\mathbf{x},t) = -\frac{1}{T_R} \log\left\{ \frac{1-s(\mathbf{x},t)+s(\mathbf{x},t)\exp[-R_1(\mathbf{x},0)T_R]-\exp[-R_1(\mathbf{x},0)T_R]\cos\theta}{1-s(\mathbf{x},t)\cos\theta+s(\mathbf{x},t)\exp[-R_1(\mathbf{x},0)T_R]\cos\theta-\exp[-R_1(\mathbf{x},0)T_R]\cos\theta} \right\}, \quad (2)$$

where $s(\mathbf{x},t)=S(\mathbf{x},t)/S(\mathbf{x},0)$ is the signal normalized at time $t=0$, θ is the flip angle (assumed constant), and T_R is the repetition time. Note that we have eliminated all terms involving $\sin\theta$. Since the error in $\sin\theta$ is larger than the error in $\cos\theta$ when θ is close to zero, eliminating $\sin\theta$ reduces sensitivity to inhomogeneities in the volume excitation (also known as $\mathbf{B}_1$ transmit radio frequency field inhomogeneities). If a B_1 field map is available, it can be used to generate an R_1 map separately that can be used as an input to `DCEMRI.jl`. Currently `DCEMRI.jl` does not support R_1 mapping with spatially varying flip angles, although nothing precludes adding that functionality in the future.

In the next stage of the processing, the effective relaxation rate $R_1(\mathbf{x},t)$ is converted to the concentration in tissue C_t of the contrast agent using

$$C_t(\mathbf{x},t) = \frac{R_1(\mathbf{x},t)-R_1(\mathbf{x},0)}{r_1},$$

where r_1 is the relaxivity of the contrast agent. For our in vivo experiment, Gd-DTPA was used, for which we take the relaxivity to be 4.5 s^{-1} mM^{-1} at 3.0 T because that was the relaxivity used in the validation data, and this same value has been found in in vivo studies Sasaki et al. (2005). This value can also be specified by the user.

Next a mask of voxels to process is required to avoid wasting time on fitting noise. If one is supplied in the input MAT file as the variable `mask`, it will override the automatic mask. Otherwise, the automatic mask is generated from a variation of the signal enhancement ratio (SER, Hylton et al., 2012), defined here as the mean signal in each voxel in the last three dynamics divided by the mean of the signal in the voxel in the first three dynamics. (Note that this requires the acquisition of three pre-contrast time points.) By default any voxels with an SER above 2.0 will be included in the processing mask. This cutoff can be changed by the user.

Tissue Models

Three main tissues models are included by default: the Standard and Extended Tofts-Kety models (Yankeelov and Gore, 2009) and a plasma-only model (no exchange limit). Other models can be added easily by the user. The Extended Tofts-Kety model is a two-compartment model that assumes that the blood vessel supplies the CA to the tissue at a slow and fixed transport rate K^{trans}. The volume of the extracellular, extravascular tissue space is labeled v_e, and the volume fraction of the blood vessels is v_p. Under this model, the tissue concentration can be written as

$$C_t(\mathbf{x},t) = K^{\text{trans}}(\mathbf{x}) \int_0^t C_p(t) \exp\left[k_{\text{ep}}(\mathbf{x})(s-t)\right] ds + v_p(\mathbf{x})C_p(t), \quad (3)$$

where the efflux rate constant

$$k_{\text{ep}}(\mathbf{x}) \equiv \frac{K^{\text{trans}}(\mathbf{x})}{v_e(\mathbf{x})}. \quad (4)$$

4/15

¹²³ Fitting the derived tissue concentration curves C_t to this model involves finding the K^{trans}, v_e, and v_p that
¹²⁴ best reproduce the observed C_t given an AIF $C_p(t)$. In the Standard Tofts-Kety model, v_p is assumed to
¹²⁵ be zero, and in the plasma-only model K^{trans} is assumed to be zero. Formulating the integral using k_{ep}
¹²⁶ instead of the ratio K^{trans}/v_e produces better fits. To get v_e, K^{trans} can be divided by k_{ep}, taking care to
¹²⁷ handle cases where $k_{ep} = 0$.

¹²⁸   The models to use are specified with a bitmask supplied in either the input MAT file or as a command
¹²⁹ line argument, and multiple models can be fit to the same voxel. The code will then choose the best
¹³⁰ fitting model based on the reduced χ^2. For each model selected, numerical integration is performed
¹³¹ using a trapezoidal rule, and the nonlinear least squares fitting is performed in parallel for each voxel
¹³² independently using the Levenberg-Marquardt method. All fitting code is written in pure Julia—no
¹³³ external libraries are called.

Postprocessing

¹³⁵ Parameters were clamped in voxels that where the fit produced unphysical values. The volume fractions
¹³⁶ v_e and v_p were clamped to the $[0,1]$ interval, while K^{trans} was clamped to the $[0,5]$ interval. The original
¹³⁷ fit residuals were retained for filtering as well. If a fit residual is large, one can safely assume that either
¹³⁸ the signal in the voxel was too low to provide an accurate fit or the model assumptions were violated
¹³⁹ at that location. In either case, poorly fitted voxels should be omitted in an imaging analysis. The user
¹⁴⁰ should select the correct, data-dependent cutoff residual, so DCEMRI.jl does not automatically filter by
¹⁴¹ residual.

¹⁴²   Finally, all results are saved to an output MAT v5 file. The name of this file can be customized through
¹⁴³ a command line argument or the variable outfile in the input MAT file.

QIBA Phantom Data

¹⁴⁵ The Quantitative Imaging Biomarkers Alliance[8] has provided virtual DCE phantoms in the DICOM format
¹⁴⁶ for validating DCE-MRI analysis codes. Several phantoms are available for benchmarking both DCE
¹⁴⁷ model fitting and T_1 mapping, with a range of noise and timing errors added. Here we chose the noise-free
¹⁴⁸ Standard and Extended Tofts phantoms (versions 6 and 4, respectively). The Standard Tofts phantom
¹⁴⁹ contains 10×10 squares of all combinations of six values of $K^{\text{trans}} \in \{0.01, 0.02, 0.05, 0.1, 0.2, 0.35\}$
¹⁵⁰ min^{-1} and five values of $v_e \in \{0.01, 0.05, 0.1, 0.2, 0.5\}$, for 30 regions total. The phantom contains 1361
¹⁵¹ time points for each voxel. The Extended Tofts-Kety phantom (version 4) contains 10×10 patches of all
¹⁵² combinations of the parameters $K^{\text{trans}} \in \{0.0, 0.01, 0.02, 0.05, 0.1, 0.2\}$ min^{-1}, $v_e \in \{0.1, 0.2, 0.5\}$, and
¹⁵³ $v_p \in \{0.001, 0.005, 0.01, 0.02, 0.05, 0.1\}$, for 108 regions total. This phantom contains 661 time points
¹⁵⁴ for each voxel. Figure 1 shows an example dynamic from the version 6 QIBA phantom along with its
¹⁵⁵ associated AIF.

¹⁵⁶   In the noisy cases, we followed Ortuño et al. (2013) and added complex Gaussian noise with standard
¹⁵⁷ deviation $\sigma = 0.2$ relative to the pre-contrast baseline to the images. We then went a step further and
¹⁵⁸ took the magnitude of the resulting data, transforming the noise distribution into the more realistic Rician
¹⁵⁹ distribution. The difference between a Gaussian and a Rician distribution is minimal for voxels with
¹⁶⁰ signal-to-noise ratios $\gtrsim 10$. No noise was added to the AIF for simplicity and to allow faithful comparisons
¹⁶¹ between this work and Ortuño et al. (2013).

¹⁶²   We extracted just one voxel from each region in the noise-free cases to reduce the computation time,
¹⁶³ since all voxels were identical in each region. We retained all 100 voxels in each region for the noisy
¹⁶⁴ cases, however, in order to sample the effects of the added noise.

In Vivo Data Collection

¹⁶⁶ In vivo breast data were acquired using a Philips[9] Achieva 3.0 Tesla MR scanner. For the T_1-weighted
¹⁶⁷ data, a 3D gradient echo multiple flip angle approach was used with $T_R = 7.9$ ms, $T_E = 1.3$ ms, and flip
¹⁶⁸ angles of 2–20 deg in 2 deg increments. Flip angles were uniformly spaced instead of optimized because
¹⁶⁹ of the broad range of tissue properties found in tumors. The acquisition matrix was $192 \times 192 \times 20$ (full
¹⁷⁰ breast) over a sagittally oriented field-of-view of 22 cm $\times$ 22 cm $\times$ 10 cm. Scan time was just under 3
¹⁷¹ min. The DCE sequence used identical parameters but with a single flip angle of 20 deg. Each 20-slice set
¹⁷² was collected in 16.5 s at 25 time points for approximately 7 min of scanning. A catheter placed within an

---

[8]https://www.rsna.org/QIBA.aspx
[9]Philips Healthcare, Best, Netherlands
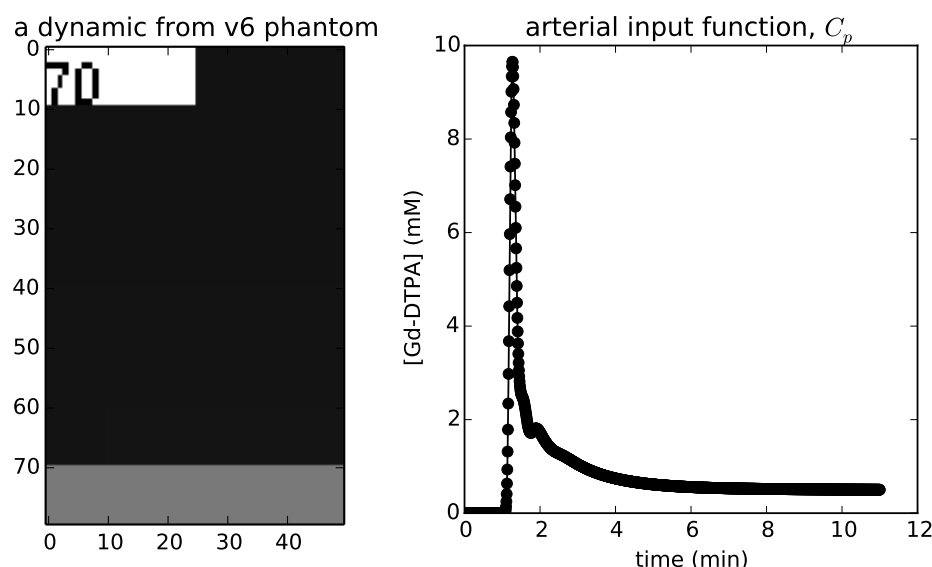
Figure 1. Example dynamic of the noise-free v6 QIBA phantom (left). The phantom contains a numeric label of its order in the time series in the upper left, the tissues regions in the middle, and a vessel strip at the bottom, from which the AIF may be extracted. The AIF extract from this phantom is shown at right.

antecubital vein delivered 0.1 mmol kg^{-1} of the contrast agent Magnevist at 2 mL s^{-1} (followed by a saline flush) via a power injector after the acquisition of three baseline dynamic scans for the DCE study. A population AIF was used Li et al. (2011).

Modes of Operation

Three modes of operation are provided for `DCEMRI.jl`. First, it can be called as a command line tool using the provided script `dcefit`. This mode is appropriate for batch processing, or as part of shell scripts or larger analysis programs written in languages other than Julia.

The second model of operation is through the supplied MATLAB interface. Results can be saved as a MAT file, and then passed to the MATLAB function `dcefit.m`. Saving a MAT file is not as fast as direct parameter passing, but the data sizes in DCE MRI are typicall small enough relative to the computational complexity of the problem that saving and reading from disk is fast compared to the total processing time.

Finally, the preferred interface is as a direct Julia module. The `DCEMRI.jl` package is built as a proper Julia module. It can be loaded with the command `using DCEMRI`, and then inside a Julia program the provided functions can be called directly. In fact, the `dcefit` command-line interface does exactly this, with some intermediate command-line argument parsing. Loading the module in an interactive Julia session will exploit the precompilation to make subsequent executions faster. For example, in our testing, the invivo demo required 47 s to run (including environment loading and writing plot files) for the first run in an interactive session, but a second complete run finished in 8.5 s. This is extremely advantageous for iterative development and batch processing, when the analysis might need to be run many times.

RESULTS AND DISCUSSION

Validation: QIBA Phantom Data

The first validation set was performed on the QIBA version 6 Standard Tofts-Kety phantom.[10] We installed `DCEMRI.jl` on a 2.4 GHz Intel Xeon E5-2665 workstation running Ubuntu 14.04.1 LTS (GNU/Linux 3.8.0-30-generic) using Julia version 0.3.1 (commit c03f413). Eight CPU workers were used. For the noise-free case with 30 voxels × 1321 time points, fitting progressed at 5.2 voxels s^{-1}, requiring 5.8 s total; for the noisy data with 3000 voxels, the fitting rate was 5.7 voxels s^{-1} and required 525 s total.

---

[10]`https://dblab.duhs.duke.edu/modules/QIBAcontent/index.php?id=1`

In the noise-free case, the recovered parameters matched the true values to within an RMS error of 0.419% for K^{trans} and 0.126% for v_e. The maximum error in the fitted parameters was 2.17% for K^{trans} and 0.570% for v_e. The concordance correlation coefficients (CCCs) were > 0.999 for K^{trans} and v_e. The fits with the largest error relative to the true value occurred in the regions with the lowest v_e and the highest K^{trans}.
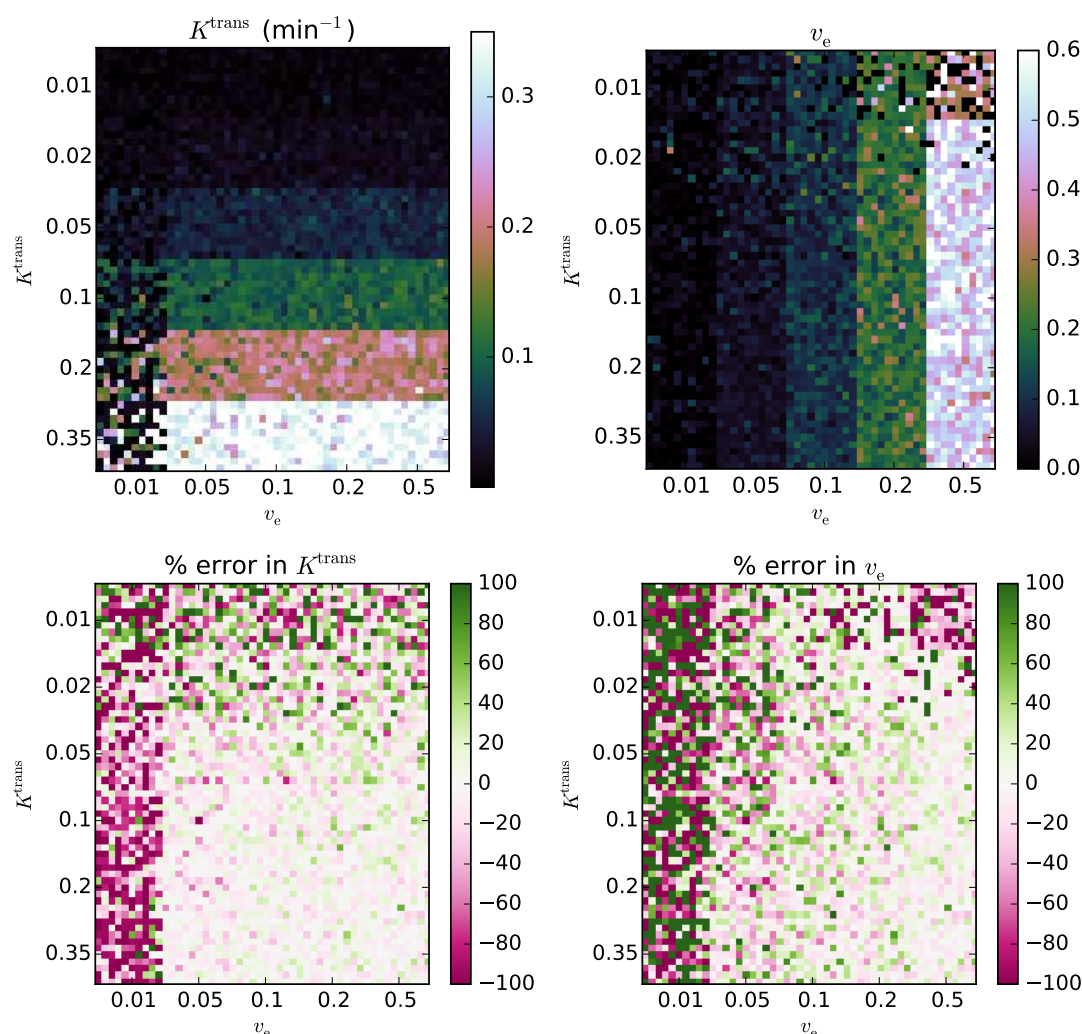
In the noisy case, using $\sigma = 0.2$, the recovered parameters agreed with the true parameters to within an RMS error of 21.5% for K^{trans} and 16.1% for v_e. The CCCs were 0.866 for K^{trans} and 0.871 for v_e. In contrast to the noise-free case, the lowest K^{trans} and v_e values had the largest relative error. The resulting parameter maps and associated errors are shown in Figs. 2 and 3.
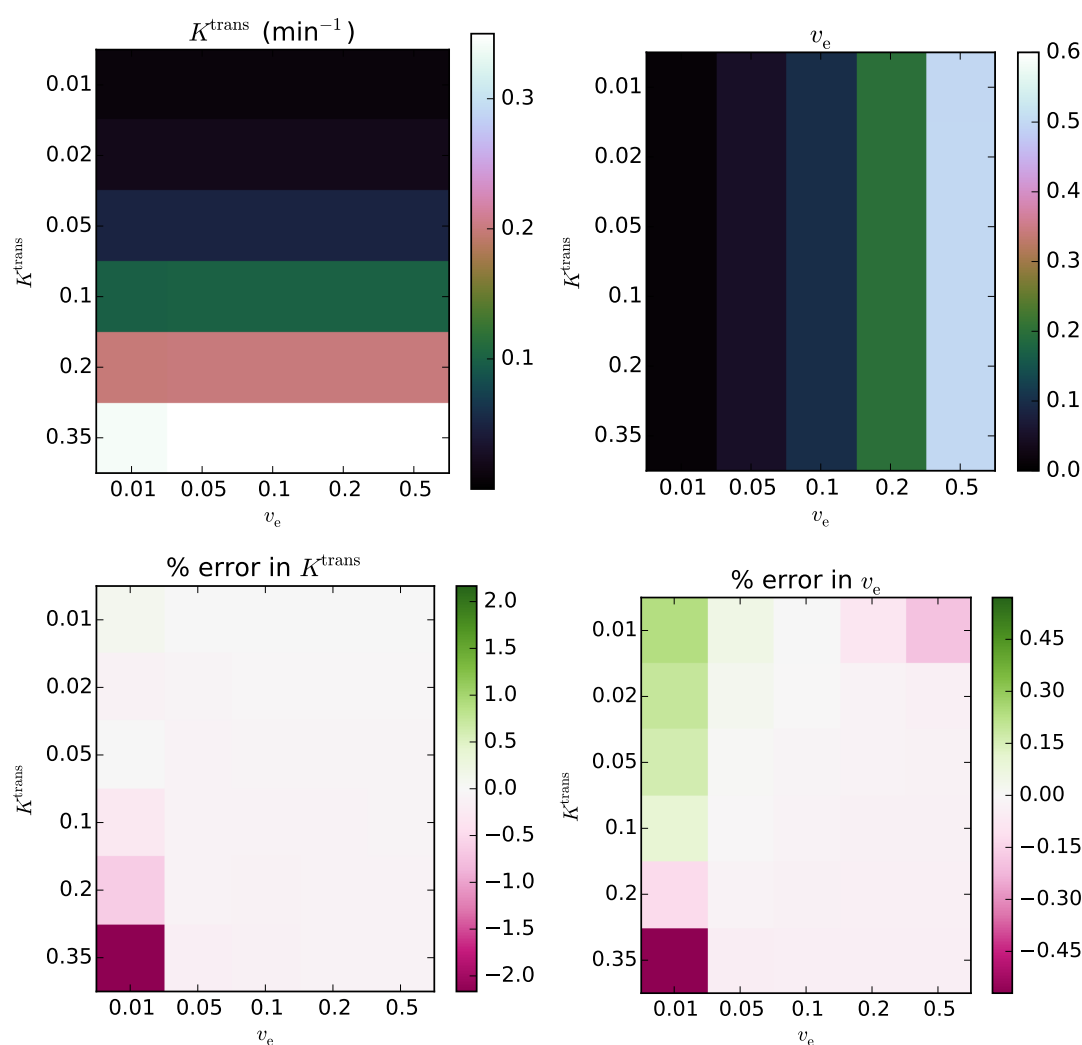
Figure 2. Fitting Standard Tofts-Kety model parameters to the QIBA v6 noise-free phantom. The RMS error was 0.419% for K^{trans} and 0.126% for v_e. CCCs were > 0.999 for both parameters.

The second validation set was performed on the QIBA version 4 Extended Tofts-Kety phantom. For this example, we removed the regions with $K^{\text{trans}} = 0$ min^{-1} from the phantom, since no transfer from the blood to the tissue violates the two-compartment model assumptions and precludes any estimation of v_e. The same software and hardware setup was used as in Validation 1. Again eight CPU workers were used. For the noise-free case with 90 regions and 661 time points, fitting progressed at 20.8 voxels s^{-1}, requiring 4.3 s total; for the noisy data with 9000 voxels, the fitting rate was 19.7 voxels s^{-1} and required 456 s total.

In the noise-free case, the recovered parameters matched the known truths to within an RMS error of 6.97% for K^{trans}, 18.0% for v_e, and 23.8% for v_p. The CCCs for both parameters were > 0.999 for K^{trans},

Figure 3. Fitting Standard Tofts-Kety parameters to the QIBA v6 phantom with $\sigma = 0.2$ noise added. The RMS error was 21.5% for K^{trans} and 16.1% for v_e; CCCs were 0.866, and 0.871, respectively.

0.890 for v_e, and > 0.999 for v_p. The fits with the largest error relative to the true values occurred in the regions with the lowest K^{trans} and, to a lesser extent, lowest v_p.

In the noisy case, using $\sigma = 0.2$, the recovered parameters agreed with the true parameters to within an RMS error of 11.3% for K^{trans}, 18.2% for v_e, and 12.7% for v_p. The CCCs were 0.974 for K^{trans}, 0.703 for v_e, and 0.972 for v_p. Against the fits with the largest relative error occurred in regions of interest with the lowest K^{trans} and v_p. The resulting parameter maps and associated errors are shown in Figs. 4 and 5.

Several factors likely contribute to the accuracy of retrieving perfusion parameters from the QIBA phantom data. Most importantly, the Jacobian of the Tofts-Kety model includes terms of the form

$$\frac{\partial C_t(t)}{\partial K^{\text{trans}}} = \int_0^t C_p(s) \exp\left[\frac{K^{\text{trans}}}{v_e}(s-t)\right] \left[1 + \frac{(K^{\text{trans}})^2}{v_e}(s-t)\right] ds, \tag{5}$$

$$\frac{\partial C_t(t)}{\partial v_e} = -\frac{(K^{\text{trans}})^2}{(v_e)^2} \int_0^t C_p(s) \exp\left[\frac{K^{\text{trans}}}{v_e}(s-t)\right](s-t) \, ds, \tag{6}$$

and

$$\frac{\partial C_t(t)}{\partial v_p} = C_p(t). \tag{7}$$

224 With such strong dependences on K^{trans} and v_{e}, the K^{trans} and v_{e} columns of the Jacobian may become
225 ill-conditioned when K^{trans} or v_{e} take on extreme values, leading to a loss of numerical precision. For
226 example, in the Standard Tofts-Kety model, the difference in dependence of the terms on v_{e} can cause
227 ill-conditioning when v_{e} is close to zero, regardless of K^{trans}, since both columns depend on K^{trans} in
228 similar ways.

229   This hypothesis is strengthened by observing that the largest error in Figs. 2 and 3 are when $v_{\text{e}} = 0.01$
230 and is roughly independent of K^{trans}. We also note that for the model assumptions to be valid v_{e} must be
231 non-zero. Thus the error increases as the parameters get closer to violating the model assumptions.

232   For the Extended Tofts-Kety model, the situation changes because the Jacobian has a different term,
233 v_{p}, that is independent of K^{trans} and v_{e}. Because of this, K^{trans} alone can now cause ill-conditioning. Figs.
234 4 and 5 are consistent with these limits. The largest error occurs for $K^{\text{trans}} = 0.01$ min^{-1} and $v_{\text{e}} = 0.5$.
235 The error in the fits can also be said to be largest when k_{ep} is small. This suggests that the numerical
236 precision of the fits should be much lower in regions of low transfer and high extravascular, extracellular
237 volume, such as the central, necrotic regions of some tumors.

238   Quantitatively, the two validation data sets recovered the parameters extremely accurately when
239 $K^{\text{trans}} \geq 0.05$ min^{-1}. Because of this we recommend caution when including voxels where $K^{\text{trans}} < 0.05$
240 in analyses.

241   As an aside, a very slight underestimation K^{trans} and v_{e} is apparent in the pinkish tint of the error
242 maps of Fig. 2, and a slight overestimation can be seen in the greenish tint of the error maps in Fig.
243 4. Neither of these effects is large when compared to the other fitting issues, however. We have no
244 explanation for this, but note it here as a point of curiosity.

Application: In Vivo Breast DCE-MRI

246 The third validation was not a test of accuracy of parameter recovery, but rather a proof of concept for
247 in vivo applications. In vivo data is more hetereogeneous and subject to measurement error and voxel
248 averaging, so not all measured voxels may follow the Standard or Extended Tofts-Kety model.

249   The same hardware and software setup was used as in Validations 1 and 2. A standard Tofts-Kety
250 model was used because of its robustness to noise and for simplicity of exposition here. `DCEMRI.jl`
251 selected 18,327 voxels as containing significant signal and created R_1 and S_0 maps in 2.9 s, for a processing
252 rate of 6365 voxels s^{-1}. Of the 18,327 voxels selected for R_1 fitting, 6774 were computed to have a signal
253 enhancement ratio of 2.0 or more and were selected for DCE model fitting. Fitting required 0.9 s, for a
254 rate of 7815 voxels s^{-1}. The resulting maps are shown in Figure 6.

255   The general features of the computed maps for the in vivo are consistent with expected results. The
256 R_1 relaxation rate is lower in the tumor than in the fatty tissue and is similar to that in the fibroglandular
257 tissue. The CA concentration is generally highest in the tumor and in vascular-like structures. The signal
258 enhancement ratio is highest in the tumor, apart from some garbage results posterior of the chess wall due
259 to breathing and cardiac motion. Finally, the derived values of K^{trans}, v_{e}, and v_{p} through the tumor are
260 consistent with typical tumor values and spatially consistent with neighboring voxels.

Run Time

262 We have collected the run times and number of time points of each of the two cases for each of the
263 two validations along with the same for the in vivo example in Fig. 7. We wanted to determine the
264 scaling of run time of `DCEMRI.jl` with problem size so that better comparisons with other packages
265 can be made. We hypothesized that the run time would be dominated by the matrix operations in the
266 Levenberg-Marquardt routine. Under this hypothesis, we assumed that a polynomial scaling of the
267 run time might occur. Since matrix multiplication can scale as poorly as $O(N^3)$, we tested low-order
268 polynomials in N and $\log N$, but we found poor fits when a zero intercept was required. A power-law
269 fit was found to fit better than polynomials, and we found that the best fit power law for the run time in
270 seconds per voxel was $t_{\text{run}}(N) = 2.2 \times 10^{-7} N^{1.9}$, where N in the number of time points per voxel.

Comparison to Other Packages

272 DCE@urLAB by Ortuño et al. (2013) found comparable errors in fitting to this work. While they didn't
273 state quantitative error measurements, their Figs. 7 and 9 were similar in character to Figs. 2–5 here.
274 They also stated run times of 20 sec to fit 1024 voxels and 40 dynamic frames and 5 minutes to fit for
275 16,384 voxels and 40 dynamic frames, or roughly 19 ms per pixel. According to our run time analysis,
276 `DCEMRI.jl` using four CPU cores would require only 0.80 ms per pixel which is 24× faster.

Also, DCE@urLAB contains around 20,000 lines of code, while `DCEMRI.jl` contains only around 1000 lines of code, and 500 of those are devoted to phantom validation and plotting. DCE@urLAB does contain more models, however, so adding additional models to `DCEMRI.jl` will require more code, but only on the order of tens of lines, not thousands.

dcemriS4 has not published validation results, so we cannot compare its accuracy to `DCEMRI.jl`, but in our own testing, we found that dcemriS4 required roughly 10 s on average to fit the Extended Tofts-Kety model to the tissue curves derived from the breast data set, while `DCEMRI.jl` required 0.9 s. This suggests that `DCEMRI.jl` may be $\sim 10\times$ faster than dcemriS4.

CONCLUSIONS

We have demonstrated an open source, free, and highly portable solution to DCE-MRI analysis that achieves similar accuracy of derived parameters, eschews needless complexity, and is 10–$20\times$ faster than comparable solutions. Many improvements are possible for `DCEMRI.jl`. First, more models can be added as long as they can be validated. Many of the existing packages include more than just the Tofts-Kety models, and `DCEMRI.jl` is written such that the model to be fit is completely independent of the fitting code itself. Thus adding new models is trivial. Second, upcoming improvements to the Julia language will bring even more speed. The planned feature of module load caching should speed up loading modules in Julia, which is currently one of the slowest parts of `DCEMRI.jl`. A major overhaul of plotting in Julia is also in progress which should improve the speed and quality of plotting. Finally, improvements in the low-level code translation to better optimize vector arithmetic on modern CPUs are in the works. As these changes are implemented, users can stay up to date simply by updating to the latest Julia stable release and then running `Pkg.update()` in the Julia environment. This command will pull the latest commits from the `DCEMRI.jl` git archive and patch the local copy of the module.

ACKNOWLEDGMENTS

REFERENCES

Bezanson, J., Edelman, A., and Karpinski, S. (2014). Julia : A fresh approach to numerical computing. pages 1–41.

Bezanson, J., Karpinski, S., Shah, V. B., and Edelman, A. (2012). Julia: A Fast Dynamic Language for Technical Computing.

Choyke, P. L., Dwyer, A. J., and Knopp, M. V. (2003). Functional tumor imaging with dynamic contrast-enhanced magnetic resonance imaging. *J Mag Reson Imag*, 17(5):509–20.

Hylton, N. M., Blume, J. D., Bernreuter, W. K., Pisano, E. D., Rosen, M. A., Morris, E. A., Weatherall, P. T., Lehman, C. D., Newstead, G. M., Polin, S., Marques, H. S., Esserman, L. J., and Schnall, M. D. (2012). Locally advanced breast cancer: MR imaging for prediction of response to neoadjuvant chemotherapy–results from ACRIN 6657/I-SPY TRIAL. *Radiology*, 263(3):663–72.

Jensen, L. R., Huuse, E. M., Bathen, T. F., Goa, P. l. E., Bofin, A. M., Pedersen, T. B., Lundgren, S., and Gribbestad, I. S. (2010). Assessment of early docetaxel response in an experimental model of human breast cancer using DCE-MRI, ex vivo HR MAS, and in vivo 1H MRS. *NMR in Biomedicine*, 23(1):56–65.

Li, X., Welch, E. B., Arlinghaus, L. R., Chakravarthy, a. B., Xu, L., Farley, J., Loveless, M. E., Mayer, I. a., Kelley, M. C., Meszoely, I. M., Means-Powell, J. a., Abramson, V. G., Grau, A. M., Gore, J. C., and Yankeelov, T. E. (2011). A novel AIF tracking method and comparison of DCE-MRI parameters using individual and population-based AIFs in human breast cancer. *Physics in medicine and biology*, 56(17):5753–69.

Lockhart, A. C., Rothenberg, M. L., Dupont, J., Cooper, W., Chevalier, P., Sternas, L., Buzenet, G., Koehler, E., Sosman, J. A., Schwartz, L. H., Gultekin, D. H., Koutcher, J. A., Donnelly, E. F., Andal, R., Dancy, I., Spriggs, D. R., and Tew, W. P. (2010). Phase I study of intravenous vascular endothelial growth factor trap, aflibercept, in patients with advanced solid tumors. *J Clin Onc*, 28(2):207–14.

Mannelli, L., Patterson, A. J., Zahra, M., Priest, A. N., Graves, M. J., Lomas, D. J., Tan, L. T., Crawford, R., Brenton, J., and Sala, E. (2010). Evaluation of nonenhancing tumor fraction assessed by

328 dynamic contrast-enhanced MRI subtraction as a predictor of decrease in tumor volume in response to
329 chemoradiotherapy in advanced cervical cancer. *Am J Roent*, 195(2):524–7.

330 Ortuño, J. E., Ledesma-Carbayo, M. J., Simões, R. V., Candiota, A. P., Arús, C., and Santos, A. (2013).
331 DCE@urLAB: a dynamic contrast-enhanced MRI pharmacokinetic analysis tool for preclinical data.
332 *BMC Bioinformatics*, 14(1):316.

333 Sasaki, M., Shibata, E., Kanbara, Y., and Ehara, S. (2005). Enhancement effects and relaxivities of
334 gadolinium-DTPA at 1.5 versus 3 Tesla: a phantom study. *Mag Reson Med Sci*, 4(3):145–9.

335 Schmid, V. J., Whitcher, B., Padhani, A. R., Taylor, N. J., and Yang, G.-Z. (2006). Bayesian Methods for
336 Pharmacokinetic Models in Dynamic Contrast-Enhanced Magnetic Resonance Imaging. *IEEE TMI*,
337 25(12):1627–1636.

338 Schmid, V. J., Whitcher, B., Padhani, A. R., Taylor, N. J., and Yang, G.-Z. (2009a). A Bayesian
339 hierarchical model for the analysis of a longitudinal dynamic contrast-enhanced MRI oncology study.
340 *MRM*, 61(1):163–74.

341 Schmid, V. J., Whitcher, B., Padhani, A. R., and Yang, G.-Z. (2009b). Quantitative analysis of dynamic
342 contrast-enhanced MR images based on Bayesian P-splines. *IEEE TMI*, 28(6):789–98.

343 Yankeelov, T. E. and Gore, J. C. (2009). Dynamic Contrast Enhanced Magnetic Resonance Imaging in
344 Oncology: Theory, Data Acquisition, Analysis, and Examples.

345 Zöllner, F. G., Weisser, G., Reich, M., Kaiser, S., Schoenberg, S. O., Sourbron, S. P., and Schad, L. R.
346 (2013). UMMPerfusion: an open source software tool towards quantitative MRI perfusion analysis in
347 clinical routine. *J Dig Imag*, 26(2):344–52.

348 Zwick, S., Strecker, R., Kiselev, V., Gall, P., Huppert, J., Palmowski, M., Lederle, W., Woenne, E. C.,
349 Hengerer, A., Taupitz, M., Semmler, W., and Kiessling, F. (2009). Assessment of vascular remodeling
350 under antiangiogenic therapy using DCE-MRI and vessel size imaging. *J Mag Reson Imag*, 29(5):1125–
351 33.
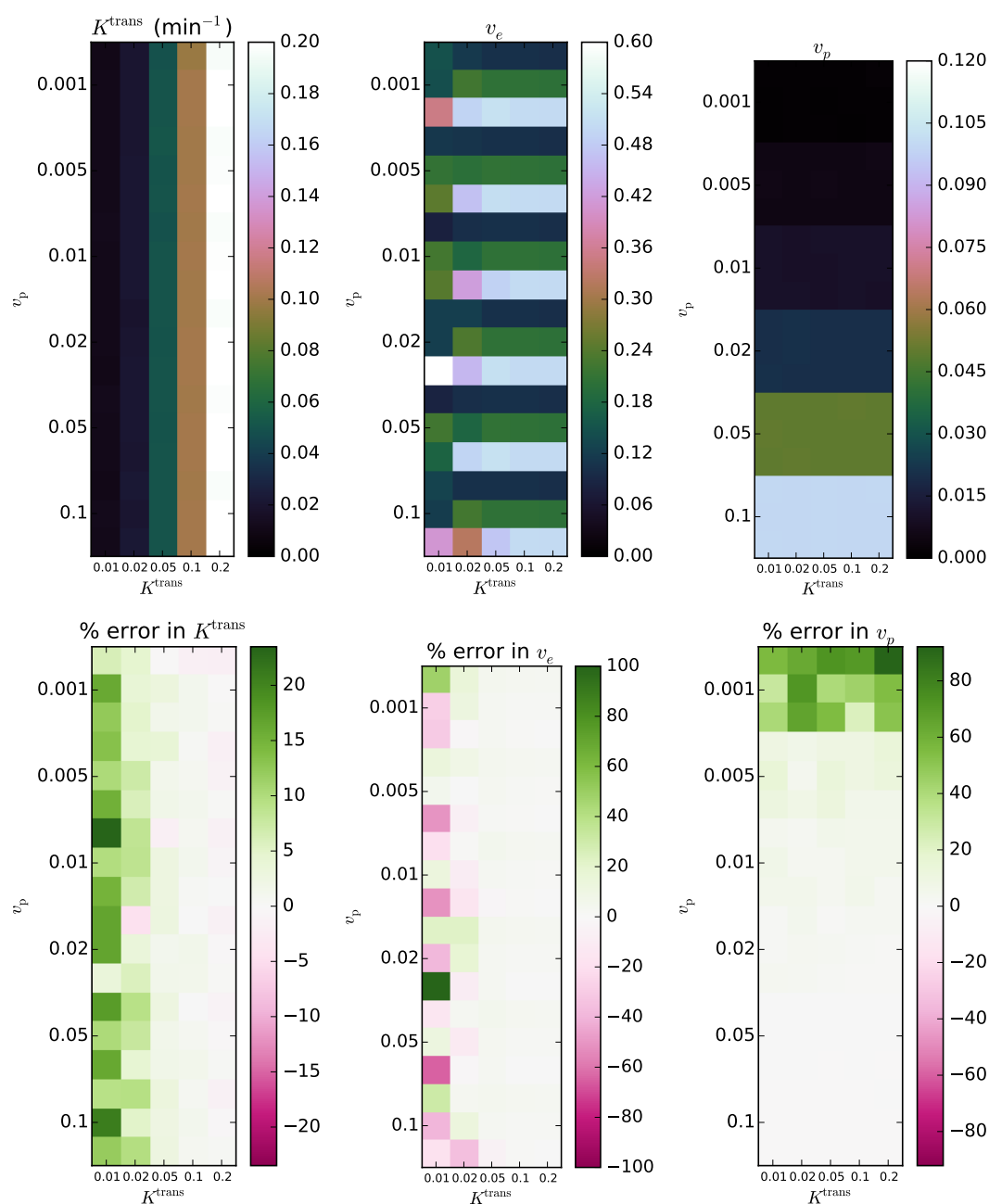
Figure 4. Fitting Extended Tofts-Kety model parameters to the noise-free QIBA v4 phantom. The RMS error was 6.97% for K^{trans}, 18.0% for v_e, and 23.8% for v_p; CCCs were > 0.999, 0.890, and > 0.999, respectively.
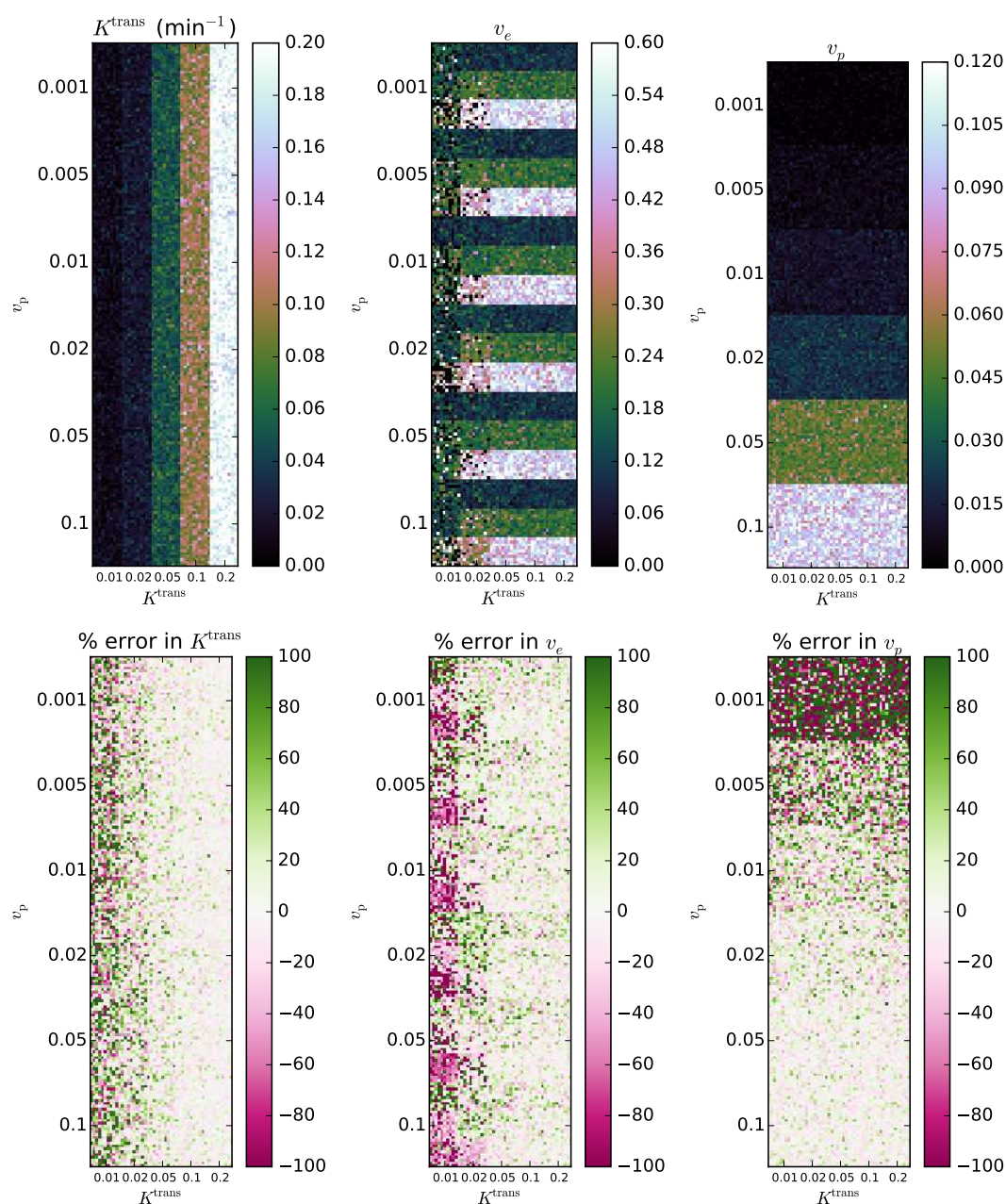
Figure 5. Fitting Extended Tofts-Kety parameters to the QIBA v4 phantom with $\sigma = 0.2$ noise added. The RMS error was 11.3% for K^{trans}, 18.2% for v_e, and 12.7% for v_p; CCCs were 0.974, 0.703, and 0.972, respectively.
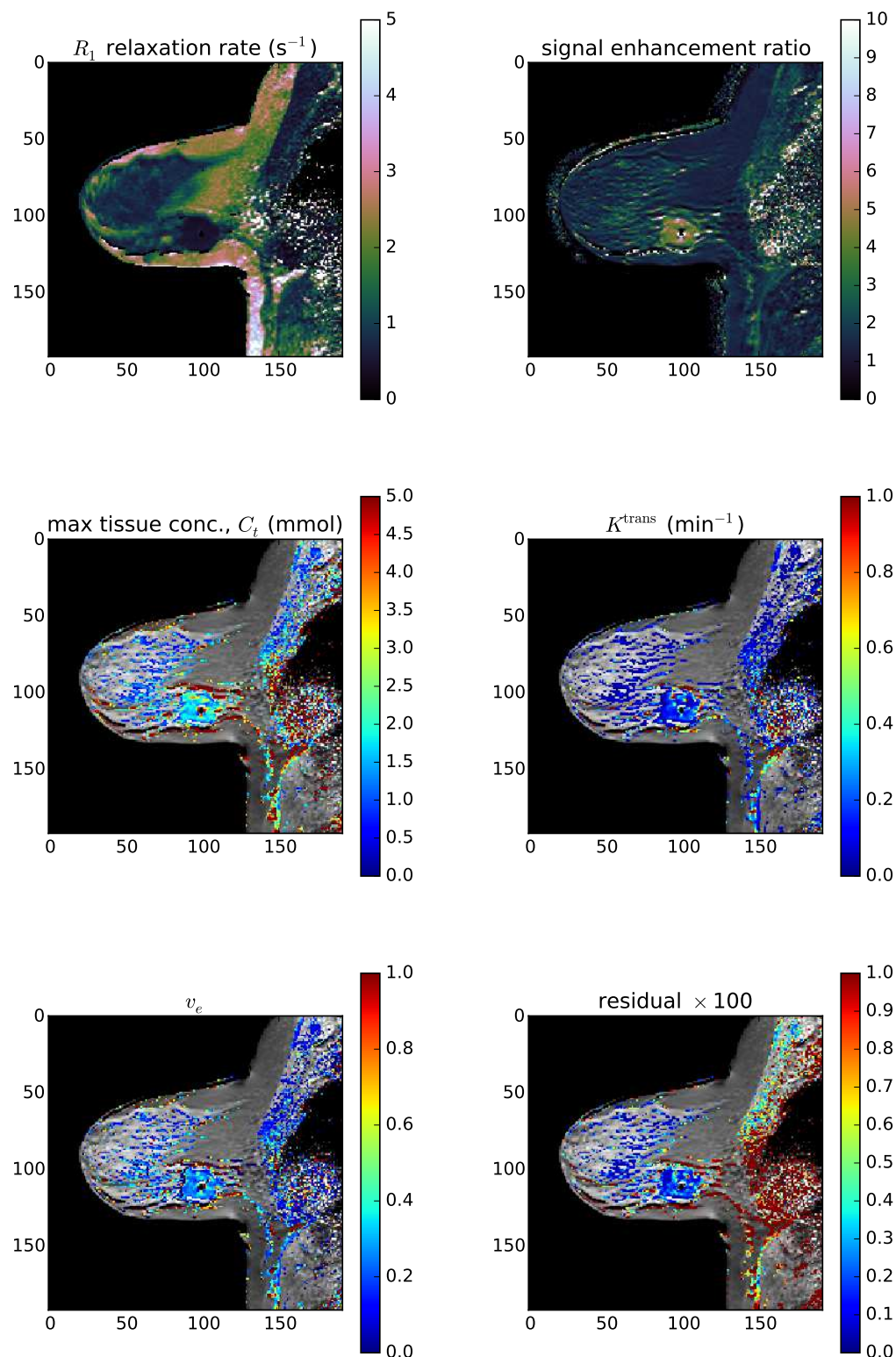
Figure 6. The computed R_1 relaxation rate, signal enhancement ratio SER, maximum CA tissue concentration C_t, the Standard Tofts-Kety parameters K^{trans} and v_e, and the fit residual. Only voxels with SER > 2.0 were fit. Of the 36,864 voxels in the image, only 7815 were selected for parameter fitting.
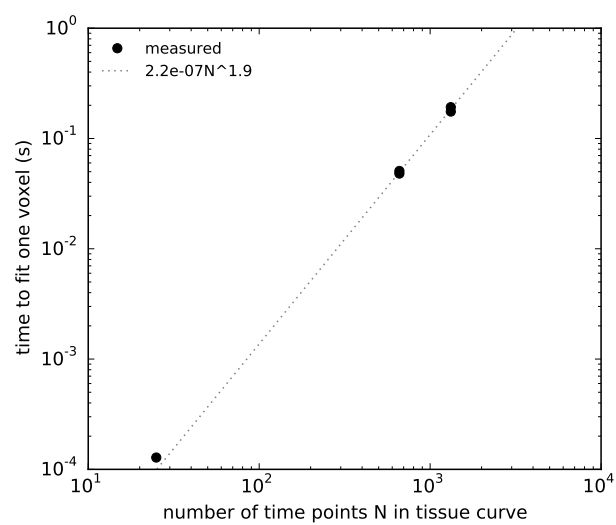
Figure 7. Time required to fit a single voxel as a function of the number of time points in the C_t curve.