A Computer Program to Extract the Continuous and Discrete Relaxation Spectra from Dynamic Viscoelastic Measurements

Arsia Takeh, Sachin Shanbhag*

Department of Scientific Computing, Florida State University, Tallahassee, FL 32306, USA

*Corresponding author: sshanbhag@fsu.edu
Fax: x1.850.644.0098

Received: 20.9.2012, Final version: 30.10.2012

Abstract:
We describe and implement an efficient, open-source, multi-platform computer program ReSpect to infer the continuous and discrete relaxation spectra from dynamic moduli measurements obtained by small-angle oscillatory shear experiments. We employ nonlinear Tikhonov regularization and the Levenberg-Marquardt method to extract the continuous relaxation spectrum. To obtain the discrete relaxation spectrum, we introduce a novel algorithm that exploits the continuous spectrum to position the modes. It uses a simple criterion which balances accuracy and conditioning of the resulting least-squares problem to determine a parsimonious number of modes. The end result is an easy-to-use, and easy-to-extend program, which can be used from the command-line or from a graphical user interface to override some of the default algorithmic choices.

Zusammenfassung:
Ziel der vorliegenden Untersuchung ist die Implementierung eines effizienten, open-source und multi-platform Computerprogramms ReSpect zur Bestimmung von kontinuierlichen und diskreten Relaxationsspektren aus dynamischen Modulmessungen, die aus kleinwinkligen oszillatorischen Schubexperimenten stammen. Die nichtlineare Tikhonov-Regularisierung sowie die Levenberg-Marquardt-Methode werden eingesetzt, um das kontinuierliche Relaxationsspektrum zu bestimmen. Dafür wird ein neuer Algorithmus eingesetzt, der auf dem kontinuierlichen Spektrum beruht, um die Positionierung der Modi anzugeben. Es wird ein einfaches Kriterium verwendet, welches einen Kompromiss zwischen Genauigkeit und Konditionierung des resultierenden Problems der kleinsten Quadrate darstellt, mit dem Ziel die genaue Anzahl an Modi zu bestimmen. Ein leicht zu bedienendes sowie erweiterbares Programm wird erstellt, welches direkt entweder von einer Befehlszeile oder von der graphischen Schnittstelle aus, einfach einige Default-Werte des Algorithmus' überschreiben kann.

Résumé:
Nous décrivons et implémentons un programme efficace, d'accès libre, multi-plateforme, appelé ReSpect, afin d'extraire les spectres de relaxation continu et discret des mesures de modules dynamiques obtenues avec des expériences de cisaillement dynamique de petite amplitude. Nous employons la régularisation non linéaire de Tikhonov et la méthode de Levenberg-Marquardt afin d'extraire le spectre de relaxation continu. Afin d'obtenir le spectre de relaxation discret, nous introduisons un nouvel algorithme qui exploite le spectre continu pour positionner les modes. Il utilise un critère simple qui balance la précision et le conditionnement du problème de moindres carrés résultant, afin de déterminer un nombre parcimonieux de modes. Le résultat final est un programme facile d'utilisation et d'extension facile, qui peut être utilisé à partir du « command-line » ou d'une interface graphique afin de surpasser certains des choix algorithmiques par défaut.

Key words: relaxation spectrum, linear rheology, continuous spectra, discrete spectra

1 INTRODUCTION

In linear rheology, the continuous relaxation spectrum (CRS), $h(\tau)$, is a quantity of vital importance, for once it is known, it is straightforward to compute all other material functions [1, 2]. Unfortunately, it cannot be measured directly; instead, it has to be inferred indirectly, most commonly, from small amplitude oscillatory shear experiments. These experiments yield the frequency-dependent dynamic moduli, $G^*(\omega) = G'(\omega) + iG''(\omega)$, where $G'(\omega)$ and $G''(\omega)$ are the storage and loss modulus, respectively, and ω is the frequency of deformation. The resulting problem of deducing $h(\tau)$ from $G^*(\omega)$ has a long and rich history [2–10]. Mathematically, $h(\tau)$ and $G^*(\omega)$ are related by:

© Appl. Rheol. 23 (2013) 24628 DOI: 10.3933/ApplRheol-23-24628

$$G'(\omega)=\int_{-\infty}^{\infty}\frac{\omega^2\tau^2}{1+\omega^2\tau^2}h(\tau)d\ln\tau$$
$$G''(\omega)=\int_{-\infty}^{\infty}\frac{\omega\tau}{1+\omega^2\tau^2}h(\tau)d\ln\tau \quad (1)$$

It is also quite common to seek a discrete relaxation spectrum (DRS) [11–14], which consists of pairs of relaxation times and strengths $\{\tau_i, g_i\}$, with $i = 1, 2, ...,N$, where N is the number of modes in the spectrum. The relationship between the DRS and $G^*(\omega)$ is given by:

$$G'(\omega)=\sum_{i=1}^{N}g_i\frac{\omega^2\tau_i^2}{1+\omega^2\tau_i^2}$$
$$G''(\omega)=\sum_{i=1}^{N}g_i\frac{\omega\tau_i}{1+\omega^2\tau_i^2} \quad (2)$$

Extracting the continous or discrete spectra is a non-trivial inverse problem in practice. Experimentally, $G^*(\omega)$ is (i) known only at discrete frequencies ω_i, (ii) restricted to a limited frequency window, and (iii) often corrupted by noise. The determination of a unique $h(\tau)$ can be complicated by these factors [15, 16]. Additional difficulties are encountered in the determination of the DRS, which is an ill-posed problem because the number of modes N (typically one per decade) is not known in advance. Furthermore, even after a suitable well-posed approximation to the problem is constructed, numerical solution is beset by problems of poor conditioning, or sensitivity to noise in the data. Despite all these issues, extracting the relaxation spectrum is an important endeavor, which partly explains the extensive body of work devoted to it.

The principal goal of this paper is to develop an open-source computer program which takes experimentally determined $G^*(\omega)$ as input, and provides the continuous and discrete relaxation spectra as output. Given the plethora of literature and computer codes already available, it is important to justify why this additional undertaking is important or useful, and how it fulfills an unmet need. Some important motivations include:

- Many of the programs based on algorithms published in the literature often reside with the original authors, and are not readily available to the general public, for use or modification [6, 9, 10, 12, 17]. Historically, it has not been easy to distribute computer code accompanying an algorithm in a primarily paper-based dissemination system. In recent years, the technical problems associated with sharing computer programs have been solved. However, there is still some understandable reluctance in sharing code because producing and maintaining well-documented and efficient code is time-consuming and not particularly rewarding [18]. It should be pointed out that while the algorithms in question are explained in sufficient detail for a trained programmer to implement, the mathematical and numerical complexities involved often put them out of reach of experimentalists who may prefer a readily available program.
- On the other end is IRIS, arguably the most popular program used in the industry for this problem (http://rheology.tripod.com). It is a commercial product whose exact underlying algorithm and implementation are, for understandable reasons, not in the public domain. Furthermore, the program currently does not run on all operating systems, and provides only the DRS, which is determined by nonlinear regression and appealing to parsimony. While the convenience afforded by such a black-box program may be completely sufficient for an experimentalist, it is less attractive to programmers who want to tinker with or extend the code, and researchers who may want to analyze particular features of the algorithm.
- In between these two extremes are powerful, freely available, well-documented, platform-independent general implementations such as DISCRETE, CONTIN, FTIKREG, NLREG, and GENEREG, which overcome many of the constraints mentioned above [3, 19–24]. These efficient programs, generally written in older versions of Fortran, extract the relaxation spectrum using some form of regularization. Paradoxically, some of the strengths of these sophisticated programs also end up being their weaknesses. For example, all of these programs are general-purpose: they can be used to not only extract $h(\tau)$ from $G^*(\omega)$, but to solve a general class of linear or nonlinear integral inverse problems. While this generality is definitely a strength, it means that solving the particular rheological problem requires one to modify the code and settings, which given the choice of the computer language and the sophisticated methods used, are an impediment to an experimentalist looking for something that works right

out of the box. In addition, many of the programs and methods listed above extract either the CRS or DRS, but not both simultaneously. They are not easily extensible, i.e, implementing a modified algorithm can be intimidating.

Thus, there is an unmet need for a computer program that potentially satisfies the following considerations simultaneously: (i) platform independence, (ii) ease-of-use for an experimentalist, (iii) transparency of the algorithm and implementation, (iv) free availability, (v) extraction of continuous and discrete relaxation spectra, (vi) efficiency (vii) readability and extensibility of the code (viii) integrated graphics (ix) extension to modern multicore machines. The computer programs mentioned above usually satisfy a subset of these design considerations. Based on these criteria, we implemented our algorithm in Matlab as the program ReSpect (after Relaxation Spectra). The same code works without any modification on the freely available "Matlab-clone" GNU Octave (http://www.gnu.org/software/octave/), which like Matlab can be installed on any operating system. This choice allows us to use several built-in numerical routines, which makes the code succinct and easy to extend.

2 METHODS

Typically, experimental $G^*(\omega)$ is available at a set of discrete frequencies ω_i, where i = 1, 2..., n, as $\{\omega_i, G_e'(\omega_i), G_e''(\omega_i)\}$. In this work, the subscript "e" denotes "experiment". We seek the continuous or discrete relaxation spectra so that the dynamic moduli implied by Equations 1 and 2 are approximately equal to the discrete $G_e^*(\omega)$.

2.1 CONTINUOUS RELAXATION SPECTRUM

In this paper, we follow the nonlinear Tikhonov regularization strategy of Honerkamp and Weese [22]. In particular, we adopt the important substitution $e^{H(\tau)} = h(\tau)$, which makes the problem nonlinear and harder to solve. However, it allows us to deal with data with large frequency range, and automatically ensures that $h(\tau)$ is positive [22]. Therefore, in this part, we seek $H(\tau)$ which minimizes the cost function given by:

$$V(\lambda)=\sum_{i=1}^{n}\left(\frac{G_e'(\omega_i)-G'(\omega_i;H(\tau))}{G_e'(\omega_i)}\right)^2+\sum_{i=1}^{n}\left(\frac{G_e''(\omega_i)-G''(\omega_i;H(\tau))}{G_e''(\omega_i)}\right)^2 +\lambda\int_{-\infty}^{\infty}\left(\frac{d^2}{d\tau^2}H(\tau)\right)^2 d\ln\tau \tag{3}$$

The first two terms on the right hand side represent the relative squared error between the experimental and inferred $G^*(\omega)$, which can be represented in shorthand as $\rho^2 = ||(G^*(H(\tau)) - G_e^*)/G_e^*||^2$. The dependence of $G^*(\omega)$ on $H(\tau)$, which is made explicit, is given by Equation 1. The last term in Equation 3 is the regularization term which consists of the regularization parameter λ, and the norm of the curvature $\eta^2 = ||d^2H(\tau)/d\tau^2||^2$.

Thus, $V(\lambda) = \rho^2 + \lambda\eta^2$. The strength of the contribution of the regularization term, which prefers smooth $H(\tau)$ can be controlled via λ. If λ is very large, the smoothness condition dominates the cost function, and we get a very smooth solution that poorly fits the experimental data (small η, and large ρ). If λ is too small, the smoothness condition is essentially ignored, and we end up with the original non-regularized problem, which is ill-conditioned. This leads to a rough $H(\tau)$ that is sensitive to noise in the data. The idea is therefore to choose some optimal $\lambda = \lambda_C$ that is midway between these extremes. There are several different methods of choosing λ_C. While the value of λ_C depends on the method used, the estimates tend to be fairly close to each other. In this program, we use the L-curve method in conjunction with a simple heuristic, but allow the user to adjust the smoothing factor manually, if needed [25–27].

It should be noted that although we formally represent $H_\lambda(\tau)$ as a function, it is represented in a discretized form in ReSpect. That is, we divide the domain between τ_{min} to τ_{max} into n_τ equally spaced grid points (on a logarithmic scale) using the relation:

$$\tau_i = \tau_{min}\left(\frac{\tau_{max}}{\tau_{min}}\right)^{\frac{i-1}{n_\tau-1}} \tag{4}$$

Typically, we set n_τ so that there are 5 - 10 grid points per decade of frequency data (In this paper, n is the number of datapoints in G_e^*, N is the number of modes in the DRS, and n_τ or n_s is the number of grid points into which the CRS is discretized). It is quite common to set $\tau_{min} = 1/\omega_{max}$ and $\tau_{max} = 1/\omega_{min}$, where ω_{min} and ω_{max} define the frequency window over which $G_e^*(\omega)$ is acquired (Strictly, the domain of τ is smaller by a factor of $\exp(\pi/2)$ on either end [15]. However, the smoothness imposed on the solution may

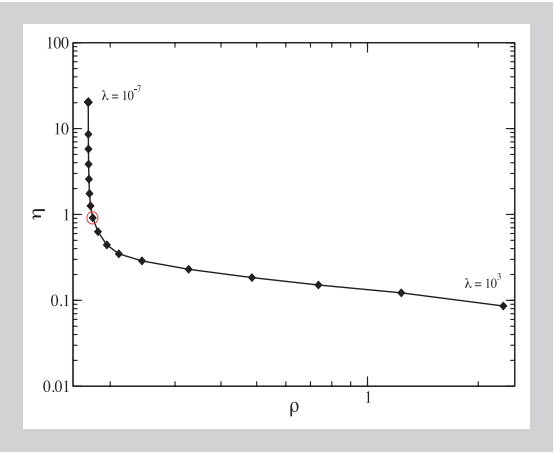


Figure 1:
This particular ρ versus η, or L-curve, was determined for the problem considered in case study 3, described in the "Results" section. As λ is increased, the monotonically-decreasing curve shows a sharp initial decline, in which η falls precipitously while ρ barely increases. This is followed by a regime, where η decreases much more gradually, while ρ increases rapidly. The "optimum" λ (open circle), determined here using a simple heuristic, lies in the corner region where the two regimes meet.

"increase" the range). Thus, using Equation 1, $G^*(\omega)$ is approximated as:

$$G'(\omega_j) = \int_{-\infty}^{\infty} \frac{\omega_j^2\tau^2}{1+\omega_j^2\tau^2} e^{H(\tau)}\, d\ln\tau \approx \Delta\log\tau \sum_{i=1}^{n_\tau} e^{H(\tau_i)} \frac{\omega_j^2\tau_i^2}{1+\omega_j^2\tau_i^2}$$
$$G''(\omega_j) = \int_{-\infty}^{\infty} \frac{\omega_j\tau}{1+\omega_j^2\tau^2} e^{H(\tau)}\, d\ln\tau \approx \Delta\log\tau \sum_{i=1}^{n_\tau} e^{H(\tau_i)} \frac{\omega_j\tau_i}{1+\omega_j^2\tau_i^2} \quad (5)$$

where $\Delta\log\tau = (\log(\tau_{max}) - \log(\tau_{min}))/(n_\tau - 1)$. For a given a value of λ, we can find the the function $H_\lambda(\tau)$ that minimizes $V(\lambda)$ using the popular Levenberg-Marquardt method [28]. We can then compute $\rho(\lambda)$ and $\eta(\lambda)$ that correspond to $H_\lambda(\tau)$ in Equation 3. In the L-curve method for finding λ_C, we repeat this step for a range of different λ. A typical plot of ρ against η is shown in Figure 1. It has a characteristic "L" shape from which the method gets its name. Depending on whether λ is small or large, the minimization process focuses on reducing either ρ or η, causing large errors in the other. In the L-curve method, some point in the "corner" of the L-shaped graph, after the steep initial decline, is used to determine λ_C. In general, this can be solved for by trying to find the point of maximum curvature on the plot. In this program, however, we use a simple heuristic to determine λ_C, viz., $\rho(\lambda_C) = 1.05\,\rho(\lambda_{min})$, where λ_{min} is the minimum λ in the range of λ explored to draw the L-curve. In Figure 1, the open circle denotes the λ_C determined by this simple heuristic. In the program, we allow the user to exercise some additional control over the specification of λ_C. Once λ_C is found, the computed CRS is given by $H_{\lambda C}(\tau)$. Thus, the algorithm used to compute $h(\tau) = \exp(H_{\lambda C}(\tau))$ from the experimental $G_e^*(\omega)$ may be summarized as:

- Pick a set of λ from small to large values (eg., $10^{-12} - 10^{13}$).
- For each λ, find the H_λ, that minimizes $V(\lambda)$ using the Levenberg-Marquardt method. Store the corresponding $\rho(\lambda)$ and $\eta(\lambda)$.
- Make a double logarithmic plot of ρ v/s η, and determine a λ_C in the corner of the L-curve.
- Compute the CRS by $h(\tau) = \exp(H_{\lambda C}(\tau))$.

2.2 DISCRETE RELAXATION SPECTRUM

We use the CRS, $h(\tau)$, to determine the DRS, $\{g_i, \tau_i\}$, $1 \le i \le N$. As mentioned before, the continuous function $h(\tau)$ is numerically represented on a computer on a dense grid as $\{h_i, \tau_i\}$, $1 \le i \le n_\tau$. Typically, $n_\tau/N \sim 10$. To avoid notational confusion between the discrete modes of the DRS τ_i, and the discrete representation of the CRS, we use the symbol s instead of τ, when referring to the CRS in this section, and reserve the symbol τ_i for the discrete modes. Thus, the CRS is denoted by $h(s) = e^{H(s)}$, and discretized as $\{h_i, s_i\}$, $1 \le i \le n_s$. A number of different strategies have been employed to determine $\{g_i, \tau_i\}$. The simplest strategy involves setting N equal to the number of decades of frequency spanned in the experimental $G^*(\omega)$, and spacing the τ_i evenly on a logarithmic scale [29, 30]. Once τ_i are thus determined, a linear least-squares problem can be set up using Equation 2. On the other extreme is the sophisticated method used in the program IRIS [12], where both – the number and placement of the modes τ_i – are treated as adjustable parameters. The resulting solution is parsimonious. However, the compactness of the spectrum can sometimes lead to features in the DRS that appear qualitatively different from the CRS.

In this work, we take an intermediate approach. Like IRIS, we let the CRS and some simple analysis guide a potentially parsimonious choice for the number and location of the $\{\tau_i\}$. However, we allow the user to incrementally overlay the simple strategy of equispaced τ_i on the parsimonious choice. The overall idea is to determine the "weight" w_j of each of s_1, ..., s_j, ..., s_{n_s} nodes of the CRS to figure out which modes can be merged or eliminated to eventually leave us with $N \ll n_s$ discrete modes. It should noted from Equation 5 that the contribution of each mode s_j is smeared across the entire frequency domain. To find the importance of s_j we need to measure its contribution to all the $G^*(\omega_i) = G'(\omega_i) + iG''(\omega_i)$, relative to the contribution of other modes s_k. As a first step, we compute the quantity:

$$w_{ij} = \frac{h(s_j)\dfrac{\omega_i^2 s_j^2}{1+\omega_i^2 s_j^2}\Delta\log s}{G'(\omega_i)} + \frac{h(s_j)\dfrac{\omega_i s_j}{1+\omega_i^2 s_j^2}\Delta\log s}{G''(\omega_i)} \quad (6)$$

where the two terms represent the contribution of the mode at s_j to $G'(\omega_i)$ and $G''(\omega_i)$, since

$$G'(\omega_i) = \Delta \log s \sum_{j=1}^{n_s} h(s_j) \frac{\omega_i^2 s_j^2}{1+\omega_i^2 s_j^2}$$
$$G''(\omega_i) = \Delta \log s \sum_{j=1}^{n_s} h(s_j) \frac{\omega_i s_j}{1+\omega_i^2 s_j^2} \quad (7)$$

The quantity w_{ij} may be thought of as the contribution of the mode s_j to $G^*(\omega_i)$. The weight w_j of the mode s_j may be obtained by summing up its contributions over all the frequencies, i.e, $w_j = \Sigma_{i=1}^{n} w_{ij}$. In ReSpect, we allow the user to "blend" this distribution of weights with a flat profile ($w_j \sim$ constant implying uniformly spaced modes) using a parameter α_f via:

$$w_j = (1-\alpha_f) \sum_{i=1}^{n} w_{ij} + \alpha_f \frac{1}{n_s} \sum_{j=1}^{n_s} \sum_{i=1}^{n} w_{ij} \quad (8)$$

The first term assigns weights to individual nodes s_j in accordance with their contribution to $G^*(\omega)$. The second term, which is constant and independent of j, corresponds to the flat profile. α_f allows us to switch smoothly between these two terms: When α_f = 0, the first term sets the weights w_j. When α_f = 1 we have a perfectly flat profile leading to equispaced discrete modes τ_i. By default, we set α_f = 0.5. We can normalize these weights, and use cubic splines to interpolate a continuous probability distribution function $\pi(s)$ through the set of normalized $\{s_j, w_j\}$. Given a particular N, we can choose the discrete modes τ_i, $1 \le i \le N$, distributed according to the density distribution $\pi(s)$. If $\pi(s)$ were uniform, the modes would be equispaced. In general, $\pi(s)$ is nonuniform resulting in a greater mode density in regions that contribute more significantly. Once the discrete modes τ_i are fixed, finding g_i is equivalent to solving the the following linear system in a least squares sense:

$$\mathbf{e} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}_{2n\times 1} \simeq \begin{bmatrix} \begin{bmatrix} \ddots & & \\ & K'_{ij} & \\ & & \ddots \end{bmatrix}_{n\times N} \\ \begin{bmatrix} \ddots & & \\ & K''_{ij} & \\ & & \ddots \end{bmatrix}_{n\times N} \end{bmatrix}_{2n\times N} \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_N \end{bmatrix}_{N\times 1} = \mathbf{Kg} \quad (9)$$

where

$$K'_{ij} = \frac{1}{G'_e(\omega_i)} \frac{\omega_i^2 \tau_j^2}{1+\omega_i^2 \tau_j^2}$$
$$K''_{ij} = \frac{1}{G''_e(\omega_i)} \frac{\omega_i \tau_j}{1+\omega_i^2 \tau_j^2} \quad (10)$$

Thus, $\mathbf{g} = (\mathbf{K}^T\mathbf{K})^{-1}\mathbf{K}^T\mathbf{e}$. We can find the condition number, cond(N), and the error by measuring $\epsilon(N) = ||(G^*_d - G^*_e)/G^*_e||^2$, where G^*_d is obtained from equation 2 on the regressed DRS. We repeat this exercise for a range of N. As N increases the error decreases, and the condition number increases (see Figure 7 for an example). The optimal number of discrete modes, N_{opt}, represents a trade-off between these two trends. In the program, we allow the user to assign the relative weights to the two quantities by specifying α_{opt}; N_{opt} is found by minimizing the cost function

$$C(N) = (1-\alpha_{opt})(\epsilon(N) - min(\epsilon(N)))^2 + \alpha_{opt} \left(\frac{cond(N)}{min(cond(N))} \right)^2 \quad (11)$$

where min($f(N)$) is the minimum value of $f(N)$ over the range of N explored. When α_{opt} = 1.0, the minimum condition number (error) determines N_{opt}. By default α_{opt} = 0.5.

3 PROGRAM

The program ReSpect can either be used directly from the command line Matlab or GNU Octave), or from a graphical user interface (Matlab only). To use ReSpect from the command line, the user specifies parameters by modifying the default settings in the function file SetParameters. These parameters include the level of discretization, the name of the input file, control of parameters such as λ_C and N_{opt}, printing, plotting etc. After setting the values of the parameters, the user calls the function contSpec() or discSpec() to get the CRS and DRS, respectively. Note that the CRS has to be computed before the computation of DRS can be attempted. In Matlab, we have also included a graphical user interface (GUI) for ReSpect. This is an alternative, and perhaps more intuitive way, of setting the parameters, and executing the programs. The GUI may be started by calling the function specgui from the Matlab command line. When the GUI is used, the parameters are not read from the SetParameters file; rather, they are specified from the GUI.

Figure 2 (left):
Filled circles denote the perturbed spectrum $H_e'(\tau)$, which is synthesized by corrupting $H_e(\tau)$ (Equation 12) with 2.5 % white noise. The solid line depicts the inferred CRS, $H'(\tau)$. The inset shows the difference between in the input and inferred spectra. Open circles correspond to the unperturbed case, where $\Delta e^{H(\tau)} = e^{H(\tau)} - e^{He(\tau)}$. Filled circles correspond to the perturbed case, where $\Delta e^{H(\tau)} = e^{H'(\tau)} - e^{He(\tau)}$.

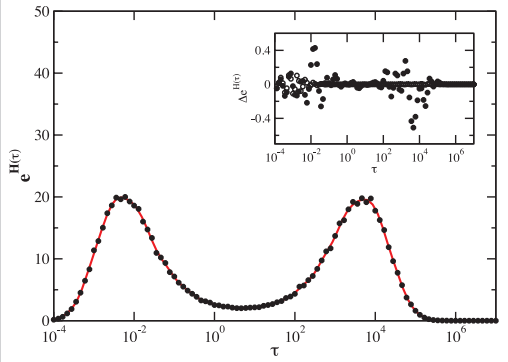
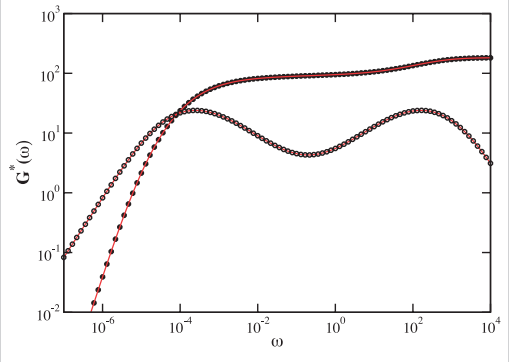


Figure 3:
The filled (unfilled) circles represent the $G'(\omega)$ ($G''(\omega)$) synthesized using $H_e'(\tau)$ ($H_e(\tau)$). The solid lines represent the corresponding dynamic modulus obtained from the corresponding CRS ($H'(\tau)$ and $H(\tau)$).



4 RESULTS

We designed three case studies to evaluate different aspects of the method and program.

- Case 1: We define a bimodal spectrum $H_e(\tau)$, and use it to synthesize $G_e^*(\omega)$, which is used as the input to the program. The goal of this case study to assess, whether the inferred $H(\tau)$ matches the input $H_e(\tau)$, reasonably well. In addition, we also study the effect of small amounts of noise in $H_e(\tau)$ on $H(\tau)$.
- Case 2: We use the Ball-McLeish theory of symmetric stars to synthesize $G_e^*(\omega)$ spanning about six decades as input. We test the quality of the inferred CRS by using it to compute the relaxation modulus $G(t)$, and comparing it with theory.
- Case 3: We use an experimental sample on binary linear-linear blends which extends over nearly seven decades as input. The goals of this case study are to (i) employ actual experimental data, and (ii) test the extraction of the DRS.

All the calculations are performed with default settings in the program, unless otherwise stated.

4.1 CASE 1: SYNTHETIC SPECTRUM

In this case study, we constructed a bimodal synthetic spectrum given by,

$$H_e(\tau) = 3 - 10^{-3} \times \left(\log\frac{\tau}{5\times10^{-3}}\right)^2 \left(\log\frac{\tau}{5\times10^{3}}\right)^2 \quad (12)$$

for τ between $\tau_{min} = 10^{-4}$ to $\tau_{max} = 10^{7}$. We discretized the domain into $n_\tau = 100$ equally spaced grid points (on a logarithmic scale) using Equation 4. We obtained the corresponding synthetic experimental data $G_e^*(\omega)$ from Equation 5, by discretizing the frequency domain into $n = 100$ logarithmically equispaced grid points w_j between $1/\tau_{max}$ to $1/\tau_{min}$. Additionally, we also perturbed the $H_e(\tau)$ in Equation 12 by adding 2.5 % white noise as $e^{He'(\tau)} = e^{He(\tau)}(1 + 0.025 \times N(0, 1))$, where $N(0, 1)$ is a normally distributed random variable with zero mean, and unit standard deviation. It is depicted in figure 2 using filled circles. The unperturbed and perturbed spectra are used to synthesize the dynamic moduli using Equation 5, which are shown in Figure 3 using symbols. For brevity, only one of the two dynamic moduli ($G'(\omega)$ or $G''(\omega)$) corresponding to $H_e(\tau)$ and $H_e'(\tau)$ is depicted.

Using ReSpect, the $G^*(\omega)$ synthesized using $H_e(\tau)$ and $H_e'(\tau)$ are used to extract the CRS, $H(\tau)$ and $H'(\tau)$, respectively. $H'(\tau)$ is denoted by the solid line in Figure 2. The difference between the inferred spectrum $e^{H(\tau)}$, and the unperturbed synthesized spectrum, $e^{He(\tau)}$, viz. $\Delta e^{H(\tau)} = e^{H(\tau)} - e^{He(\tau)}$ is shown in the inset. It also depicts $e^{H'(\tau)} - e^{He(\tau)}$. For the unperturbed case, the inferred and synthesized spectra essentially lie on top of each other. For the perturbed case, while there is some expected noise in the fit, from the Figures 2 and 3 it is clear that the extracted spectrum provides a good smooth approximation to the input spectra.

4.2 CASE 2: SYMMETRIC STAR

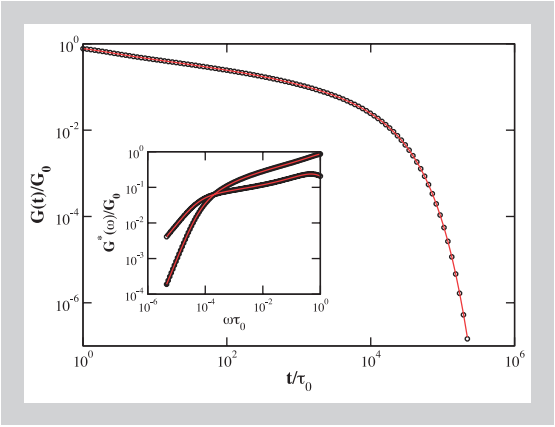
In this case, we use the Ball-McLeish dynamic dilution theory of symmetric stars to synthesize the input $G^*(\omega)$ [31]. We consider a symmetric star with $Z = M/M_e = 20$ entanglements, where M and M_e are the arm molecular weight, and the entanglement molecular weight, respectively. According to the Ball-McLeish theory, $G^*(\omega)$ for a symmetric star is given by:

$$\frac{G'(\omega)}{G_o} = \frac{2}{Z}\int_{\xi=o}^{Z} d\xi\left(1-\frac{\xi}{Z}\right)\frac{\omega^2\tau^2(\xi)}{1+\omega^2\tau^2(\xi)}$$
$$\frac{G''(\omega)}{G_o} = \frac{2}{Z}\int_{\xi=o}^{Z} d\xi\left(1-\frac{\xi}{Z}\right)\frac{\omega\tau(\xi)}{1+\omega^2\tau^2(\xi)} \quad (13)$$

where

$$\tau(\xi) = \tau_o^{\left[\frac{2v}{Z}\left(\frac{\xi^2}{2}-\frac{\xi^3}{3Z}\right)\right]} \quad (14)$$

and ξ is a contour length variable that goes from 0 at the arm tips to Z at the branching point. τ_o and G_o are the equilibrium relaxation time of a Rouse segment, and the plateau modulus,



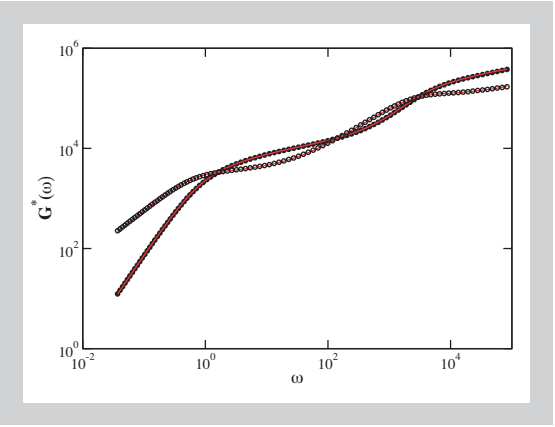
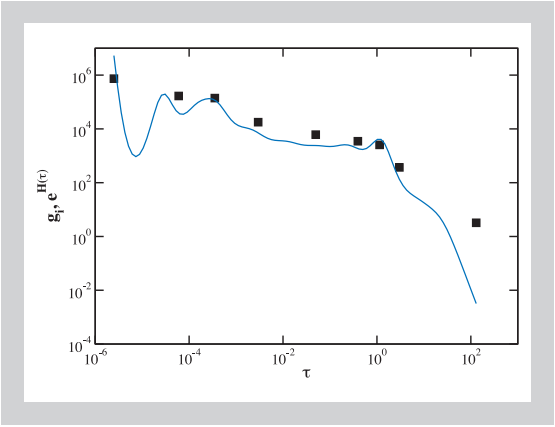


Figure 4 (left above): Symbols and lines depict the G(t) obtained using Equation 16 and 15, respectively. In the inset, the symbols denote the input G(ω) synthesized using the Ball-McLeish theory for symmetric stars, and the lines denote the G*(ω) implied by the inferred CRS.*



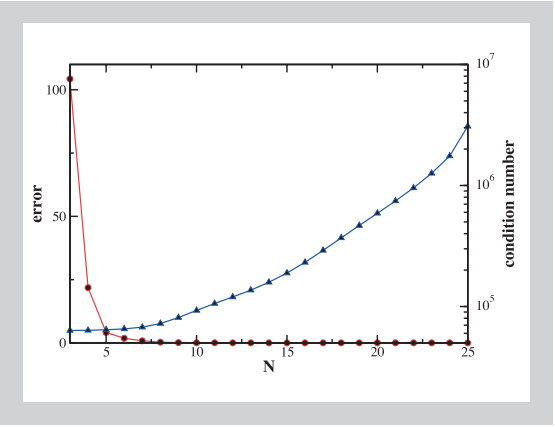


Figure 5 (right above): Symbols represent the experimental G'(ω) (filled) and G''(ω) (unfilled) for a binary linear-linear polyisoprene blend from Watanabe et al [35]. The solid lines represent the corresponding dynamic modulus obtained from the corresponding inferred spectrum. The lines for G'(ω) and G''(ω) are obtained from the inferred CRS and DRS, respectively.

Figure 6 (left below): The line is the CRS h(τ), and the squares represent the discrete relaxation spectra with N = 9 modes.

Figure 7 (right below): Plot to determine N_{opt}, the optimum number of discrete modes. The default criterion used in the program yielded N_{opt} = 9. The relative error ϵ(N) between the input data and the G(ω) inferred from the DRS is plotted on the left y-axis, while the condition number of the resulting linear least squares problem is plotted on a logarithmic scale on the right y-axis.*

respectively. The ν is the so-called prefactor of the quadratic retraction potential [32–34]. We used the $G^*(\omega)$ generated using Equations 13 and 14 with $Z = 20, \tau_o = 1, G_o = 1, \nu = 1.5$ as input, and recovered the relaxation spectrum $h(\tau)$ using ReSpect. We compared the relaxation time modulus $G(t)$ obtained from $h(\tau)$

$$\frac{G(t)}{G_o} = \int_{-\infty}^{\infty} h(\tau)^{\left(-\frac{t}{\tau}\right)} d\ln\tau \tag{15}$$

with the $G(t)$ given by the Ball-McLeish theory as:

$$\frac{G(t)}{G_o} = \frac{2}{Z}\int_{s=o}^{Z} ds\left(1-\frac{s}{Z}\right)^{\left(-\frac{t}{\tau(\xi)}\right)} \tag{16}$$

to obtain Figure 4 which shows excellent agreement.

4.3 CASE 3: A BINARY LINEAR-LINEAR BLEND

In this case study, we consider a particular polyisoprene linear-linear binary blend from Watanabe et al [35]. The particular dataset considered here is the 21K–308K, ϕ = 0.200 sample. The raw data is presented in Figure 5 as symbols. The CRS $h(\tau)$ is extracted using default settings in the program (depicted in Figure 6 as a solid line). The L-curve corresponding to this dataset was presented earlier in figure 1, which led to an optimum $\lambda_C = 1.33 \times 10^{-4}$. We used $h(\tau)$ to extract the DRS. When we scanned a range of N to determine the optimal number of modes, N_{opt}, we obtained Figure 7. As expected the error decreases with increasing N, since it is easier to fit data when a large number of free parameters are available. At the same time, the condition number, which measures the ill-conditioning of the resulting linear least squares problem, increases with N. A large N is desirable for greater accuracy, while a small N is desirable from the standpoint of conditioning. Using the default heuristic criteria, the program determines $N_{opt} = 9$ as the optimal number of modes. This number can be manually overridden in the program, either by changing α_{opt} in the optimality criterion (Equation 11), or by explicitly specifying N_{opt}. In Figure 6, we plot the DRS using N = 9, along with the CRS. The overall shapes of the CRS and DRS seem reasonable. The $G^*(\omega)$ predicted by the inferred spectra are depicted as lines in Figure 5.

4.4 COMPUTATION TIME

The extraction of the DRS from $h(\tau)$ is quite fast – it takes less than 1 second on a single modern desktop processor for all the three cases considered here. The extraction of $h(\tau)$ is more time-consuming, taking between 1–10 seconds, depending on the data, platform, printing and graphing settings, level of discretization used etc. We compared the execution time using Mat-

Table 1: Benchmark execution times for extracting h(τ) using Matlab and GNU Octave. Times are reported in seconds.

Case	Matlab	Octave
1	1.4	3.0
2	2.3	6.5
3	42	11.0

lab R2011b and GNU Octave 3.4.2 on a 2.53 GHz desktop running Red Hat Enterprise Linux 6.2 (on Linux kernel 2.6), using default settings. The results are presented in table 1.

On Matlab, the calculation is about two times faster than it is on GNU Octave, although the speed of the program on either platform is currently adequate for most purposes. About half of the execution time was spent in computing λ_C on either platform, while the rest was spent on reading, initializing, post-processing, and printing. This operation, which is single instruction multiple data (SIMD), can be easily generalized in Matlab to exploit multicore architectures, using the parfor command. Simply replacing the main for loop in the function lcurve (which scans a range of λ to determine λ_C) with parfor cuts the computation time by a factor inversely proportional to the number of cores. Thus, on a quad-core desktop, the computing time can be reduced to under a second simply by changing one word.

5 SUMMARY AND PERSPECTIVE

In this work, we provided the motivation for a free, fast, open-source, multi-platform program to extract the continuous and discrete relaxation spectra from dynamic moduli $G^*(\omega)$ obtained by small-angle oscillatory shear experiments. We introduced and described the algorithms underlying the program ReSpect, which satisfies many of these requirements. It runs on Matlab and GNU Octave and is easy-to-use for an experimentalist (especially the GUI) and easy-to-modify for a developer. Some of the methods (eg. determination of λ_C, N_{opt}, and the placement of τ_i) used in the current version of the program are heuristic. While they appear to work reasonably well in practice, many of the choices, like most heuristics, are somewhat subjective. To compensate, the program allows the user to modify or override the default choices made by the program.

In the future, we anticipate building additional modules based on published algorithms into ReSpect. For example, instead of a single heuristic algorithm to determine λ_C, the user will have a choice of the SC-method [23], the point of maximum curvature etc [26]. There are also many desirable features in the extraction of the DRS that are currently missing. Besides more algorithms to choose the number and spacing of the discrete modes, improvements can be made to the actual solution of the resulting linear least squares problem. In addition, we need to incorporate additional tools for error analysis and preprocessing experimental data. To facilitate collaboration, we expect to host ReSpect on Matlab Central (http://www.mathworks.com/matlabcentral/fileexchange/). This would attract a larger audience and greater feedback, which would help develop improved versions of the program.

6 APPENDIX: SHORT MANUAL

Unzip the folder ReSpect.zip, to obtain a sample $G^*(\omega)$ (Gst.dat), and a set of M-files. For a user using the command line interface, there are two programs, contSpec and discSpec, which can be called to obtain the continuous or discrete spectra, respectively. To adjust the default settings, the user is required to modify the settings in the function SetParameters. The GUI (only Matlab) is a simple add-on which enables the user to interact with these three files (SetParameters, contSpec and discSpec) via an intuitive graphical interface.

6.1 OVERALL LAYOUT

The program contSpec reads in the experimental data (Gst.dat) and the parameters, and initializes a CRS using InitializeH. It then samples a range of λ using lcurve. At each λ it computes the $H_\lambda(\tau)$ by using the Levenberg- Marquardt algorithm (LevenMarq). LevenMarq uses the function kernel to compute the $G^*(\omega)$ corresponding to a trial $H(\tau)$ via Equation 5, and the function kernelD, which computes the Jacobian. The function lcurve then considers the $\rho - \eta$ curve, and picks an optimal λ_C. The program finally returns λ_C and the $H_{\lambda C}(\tau)$.

The program discSpec reads in the experimental data and the computed $H_{\lambda C}(\tau)$, and computes the weights w_j of the nodes (Equation 8) using GetWeights. It then samples a range of N to determine N_{opt}. At each N, it uses the weights w_j to determine τ_j (GridDensity). The function MaxwellModes solves the resulting linear least

squares problem to obtain the discrete spectrum. After determining N_{opt} using a heuristic which tries to balance conditioning and accuracy, the function PlotMaxwellModes does some basic plotting.

There are only two functions that are shared by contSpec and discSpec, viz. kernel and SetParameters. All the data files (with suffix '.dat'), except the input file Gst.dat, are results of computation written by the software. Other than specgui.m, the other files related to the GUI are specgui.fig, which sets the basic background, and updateplot which updates plots in the figure window of the GUI, when the user selects a new or different plot.

6.2 EXAMPLE

Using the default settings, we call the function: >> contSpec(): If the function is called without any input arguments, it reads the parameters from the file SetParameters. Alternatively, it is possible to call the function by specifying the parameters (contSpec(par)), in which case the settings in the file SetParameters are not used. contSpec computes $H_{\lambda C}(\tau)$, writes it to the file H.dat, and displays it graphically. It also computes the corresponding $G^*(\omega)$, writes it to the file Gfit.dat, and displays the fit with the input data graphically. It writes the range of λ sampled, with the corresponding ρ and η in the file rho-eta.dat. >> discSpec(): Like contSpec, it reads parameters from the file SetParameters, when called without any input arguments. discSpec also computes the corresponding $G^*(\omega)$, writes it to the file Gfitd.dat, and displays the fit with the input data graphically. It writes the range of N sampled and the corresponding error and condition numbers in the file Nopt.dat.

REFERENCES

[1] Ferry J: Viscoelastic Properties of Polymers, 3rd edition, John Wiley & Sons Inc, Hoboken, NY (1980).

[2] Dealy JM, Larson RG: Molecular Structure and Rheology of Molten Polymers, Hanser Publications (2006).

[3] Honerkamp J, Weese J: Determination of the relaxation spectrum by a regularization method, Macromolecules 22 (1989) 4372–4377.

[4] Orbey N, Dealy JM: Determination of the relaxation spectrum from oscillatory shear data, J. Rheol. 35 (1991) 1035–1049.

[5] Baumgaertel M, Derosa ME, Machado J, Masse M, Winter HH: The relaxation-time spectrum of nearly monodisperse polybutadiene melts, Rheol. Acta 31 (1992) 75–82.

[6] Mead DW: Determination of molecular-weight distributions of linear flexible polymers from linear viscoelastic material functions, J. Rheol. 38 (1994) 1797–1827.

[7] Winter HH: Analysis of dynamic mechanical data: Inversion into a relaxation time spectrum and consistency check, J. Non-Newtonian Fluid Mech. 68 (1997) 225–239.

[8] Brabec CJ, Rogl H, Schausberger A: Investigation of relaxation properties of polymer melts by comparison of relaxation time spectra calculated with different algorithms, Rheol. Acta 36 (1997) 667–676.

[9] Anderssen R, Davies AR: Simple moving-average formulae for the direct recovery of the relaxation spectrum, J. Rheol. 45 (2001) 1–27.

[10] Stadler F, Bailly C: A new method for the calculation of continuous relaxation spectra from dynamic-mechanical data, Rheol. Acta 48 (2009) 33–49.

[11] Ronca G: Frequency spectrum and dynamic correlations of concentrated polymer liquids, J. Chem. Phys. 79 (1983) 1031–1043.

[12] Baumgaertel M, Winter HH: Determination of discrete relaxation and retardation time spectra from dynamic mechanical data, Rheol. Acta 28 (1989) 511–519.

[13] Baumgaertel M, Winter H: Interrelation between continuous and discrete relaxation time spectra, J.Non-Newtonian Fluid Mech. 44 (1992) 15–36.

[14] Jensen EA: Determination of discrete relaxation spectra using simulated annealing, J. Non-Newtonian Fluid Mech. 107 (2002) 1–11.

[15] Davies AR, Anderssen RS: Sampling localization in determining the relaxation spectrum, J. Non-Newtonian Fluid Mech. 73 (1997) 163–179.

[16] Macdonald JR: On relaxation-spectrum estimation for decades of data: Accuracy and sampling-localization considerations, Inverse Probl. 16 (2000) 1561–1583.

[17] Hansen S: Estimation of the relaxation spectrum from dynamic experiments using Bayesian analysis and a new regularization constraint, Rheol. Acta 47 (2008) 169–178.

[18] Ince D, Hatton L, Graham-Cumming J: The case for open computer programs, Nature 482 (2012) 485–488.

[19] Provencher SW: An eigenfunction expansion method for the analysis of exponential decay curves, J. Chem. Phys. 64 (1976) 2772–2777.

[20] Provencher SW: Contin: A general purpose constrained regularization program for inverting noisy linear algebraic and integral equations, Comp. Phys. Comm. 27 (1982) 229–242.

[21] Weese J: A reliable and fast method for the solution of fredholm integral equations of the first

kind based on Tikhonov regularization, Comp. Phys. Comm. 69 (1992) 99–111.
[22] Honerkamp J, Weese J: A nonlinear regularization method for the calculation of relaxation spectra, Rheol. Acta 32 (1993) 65–73.
[23] Weese J: A regularization method for nonlinear ill-posed problems, Comp. Phys. Comm. 77 (1993) 429–440.
[24] Roths T, Marth M, Weese J, Honerkamp J: A generalized regularization method for nonlinear ill-posed problems enhanced for nonlinear regularization terms, Comp. Phys. Comm. 139 (2001) 279–296.
[25] Hansen P: Analysis of discrete ill-posed problems by means of the Lcurve, SIAM Rev. 34 (1992) 561–580.
[26] Hansen PC, O'Leary DP: The use of the l-curve in the regularization of discrete ill-posed problems, SIAM J. Sci. Comp. 14 (1993) 1487–1503.
[27] Johnston P, Gulrajani R: Selecting the corner in the L-curve approach to Tikhonov regularization, IEEE Trans. Biomed. Engg. 47 (2000) 1293–1296.
[28] Heath M: Scientific Computing: An Introductory Survey, McGraw-Hill (2002).
[29] Emri I, Tschoegl NW: Generating line spectra from experimental responses. Part I: Relaxation modulus and creep compliance, Rheol. Acta 32 (1993) 311–322.
[30] Kaschta J, Stadler F: Avoiding waviness of relaxation spectra, Rheol. Acta 48 (2009) 709–713.
[31] Ball RC, Mcleish TCB: Dynamic dilution and the viscosity of star polymer melts, Macromolecules 22 (1989) 1911–1913.
[32] Doi M, Kuzuu NY: Rheology of star polymers in concentrated-solutions and melts, J. Polym. Sci. C-Polym. Lett. 18 (1980) 775–780.
[33] Shanbhag S, Larson RG: Chain retraction potential in a fixed entanglement network, Phys. Rev. Lett. 94 (2005) 076001.
[34] Shanbhag S, Larson RG: Identification of topological constraints in entangled polymer melts using the bond-fluctuation model, Macromolecules 39 (2006) 2413–2417.
[35] Watanabe H, Ishida S, Matsumiya Y, Inoue T: Test of full and partial tube dilation pictures in entangled blends of linear polyisoprenes, Macromolecules 37 (2004) 6619–6631.