Received: 9 October 2023 | Revised: 16 January 2024 | Accepted: 4 March 2024

DOI: 10.1002/nme.7481

REVIEW ARTICLE

WILEY

On sparse regression, L_p-regularization, and automated model discovery

Jeremy A. McCulloch[1] | **Skyler R. St. Pierre**[1] | **Kevin Linka**[2] | **Ellen Kuhl**[1]

[1]Department of Mechanical Engineering and Bioengineering, Stanford University, Stanford, California, USA

[2]Institute for Continuum and Material Mechanics, Hamburg University of Technology, Hamburg, Germany

Correspondence
Ellen Kuhl, Department of Mechanical Engineering and Bioengineering, Stanford University, 452 Escondido Mall, Stanford, CA 94305, USA.
Email: ekuhl@stanford.edu

Funding information
Division of Civil, Mechanical and Manufacturing Innovation, Grant/Award Number: CMMI-2320933; Deutscher Akademischer Austauschdienst; National Science Foundation

Abstract

Sparse regression and feature extraction are the cornerstones of knowledge discovery from massive data. Their goal is to discover interpretable and predictive models that provide simple relationships among scientific variables. While the statistical tools for model discovery are well established in the context of linear regression, their generalization to nonlinear regression in material modeling is highly problem-specific and insufficiently understood. Here we explore the potential of neural networks for automatic model discovery and induce sparsity by a hybrid approach that combines two strategies: regularization and physical constraints. We integrate the concept of L_p regularization for subset selection with constitutive neural networks that leverage our domain knowledge in kinematics and thermodynamics. We train our networks with both, synthetic and real data, and perform several thousand discovery runs to infer common guidelines and trends: L_2 regularization or ridge regression is unsuitable for model discovery; L_1 regularization or lasso promotes sparsity, but induces strong bias that may aggressively change the results; only L_0 regularization allows us to transparently fine-tune the trade-off between interpretability and predictability, simplicity and accuracy, and bias and variance. With these insights, we demonstrate that L_p regularized constitutive neural networks can simultaneously discover both, interpretable models and physically meaningful parameters. We anticipate that our findings will generalize to alternative discovery techniques such as sparse and symbolic regression, and to other domains such as biology, chemistry, or medicine. Our ability to automatically discover material models from data could have tremendous applications in generative material design and open new opportunities to manipulate matter, alter properties of existing materials, and discover new materials with user-defined properties.

KEYWORDS

automated model discovery, constitutive modeling, hyperelasticity, Lp regularization, sparse regression

This manuscript is dedicated to Robert L. Taylor on the occasion of his 90th birthday and to his many contributions to the finite element method and his open source software FEAP. Congratulations, Bob: tang,, 1!

1 | MOTIVATION

The ability to discover meaningful constitutive models from data would forever change how we understand, model, and design new materials and structures. Massive advancements in data science are now bringing us closer than ever towards this goal.[1,2] Throughout the past three years, numerous research groups have begun to harness the potential of neural networks and fit constitutive models to experimental data,[3–14] an approach that is now widely known as *constitutive neural networks*.[15] While initial studies have used neural networks exclusively as black box regression operators,[16] recent approaches are increasingly recognizing their potential to discover not only the model parameters, but also the model itself.[17] The paradigm of *automated model discovery* was first formalized in the context of nonlinear dynamical systems more than a decade ago to discover Lagrangians and Hamiltonians for oscillators,[18] pendula, biological processes,[19] or turbulent fluid flows.[20] It is now rapidly gaining popularity in the context of constitutive modeling,[21] and several promising techniques have emerged to decipher constitutive relations between stresses and strains,[11] and even integrate them automatically into a finite element analysis.[22–24] These not only include constitutive neural networks,[15] but also sparse regression,[25] genetic programming in the form of symbolic regression,[26] and variational system identification.[27]

The holy grail in automated model discovery is to identify *generalizable* and truly *interpretable* models with *physically meaningful* parameters.[2,28,29] Ideally, we want to discover a concise, yet simple and interpretable model with only a few relevant terms that best explains experimental data, while remaining robust to outliers and noise. In terms of statistical learning, this translates model discovery into a *subset selection* or feature extraction task. Subset selection and shrinkage methods are by no means new; in fact, they have been extensively studied for many decades.[30–33] In the context of linear regression, these methods have become standard textbook knowledge.[34] In the context of nonlinear regression, when analytical solutions are rare, subset selection is much more nuanced, general recommendations are difficult, and feature extraction becomes highly problem-specific.[35] To be clear, this limitation is not exclusively inherent to automated model discovery with constitutive neural networks—it applies to distilling scientific knowledge from data in general.[19] This includes alternative model discovery approaches like sparse regression,[20,25] or symbolic regression.[19,26,36] The key question to the success of discovering new knowledge from data is: how do we robustly discover the best interpretable model with a small subset of relevant terms? And, probably equally importantly: What is the trade-off between *interpretability* and prediction *accuracy*? To frame these questions more broadly, let us first revisit the notions of regression and neural networks in the context of constitutive modeling:

Regression. Regression is a statistical method to examine the relationship between a *dependent variable*, in constitutive modeling in solid mechanics the stress $\boldsymbol{\sigma}$, and one or more *independent variables*, in this case the strain $\boldsymbol{\varepsilon}$, using a model that depends on a set of *model parameters* $\boldsymbol{\theta}$. Here regression has two main objectives: characterizing the form and strength of the relationship between stress and strain to enable predictions, and providing insights into how stress and strain are correlated.[34] Popular types of regression are *logistic regression*, assuming for example a binary relationship; *linear regression*,[37] assuming a relationship that is linear in the model parameters $\boldsymbol{\theta}$, or *nonlinear regression*, assuming a relationship that is nonlinear in the model parameters $\boldsymbol{\theta}$, as we do throughout this manuscript. Regression is the cornerstone of statistical learning.[35] It provides tools to decipher relationships within data, but its application to constitutive modeling requires attention to physical constraints including objectivity, symmetry, incompressibility, polyconvexity, or thermodynamic consistency.[38–43] As a natural consequence, we cannot just use *any* set of functions to build our constitutive model: while polynomial functions between stresses and strains associated with a linear regression would be ideal from an optimization point of view, these models may violate thermodynamic constraints, which favor exponential or power functions associated with a nonlinear regression.

Linear regression. Linear regression[37] seeks to model the relationship between a *dependent variable*, in our case the stress σ, and a set of one or more *independent variables*, in our case the set of strains ε_i at different load levels i, using a function that depends *linearly* on the *model parameter* $\theta = \{ \mathrm{E} \}$, in this case the elastic modulus or Young's modulus. The regression estimates this parameter by minimizing the difference between the predicted stress, $\sigma_i = \mathrm{E}\ \varepsilon_i$, for given strains ε_i and stiffness E, and the experimentally measured stresses $\hat{\sigma}_i$, divided by the number of data points n_{data}. A common

measure for this difference is the mean squared error based on the L_2-norm,[34] $||(\circ)|| = ||(\circ)||_2 = |(\circ)^2|^{1/2}$, for which the minimization problem becomes

$$L(\mathrm{E};\varepsilon) = \frac{1}{n_{\text{data}}}\sum_{i=1}^{n_{\text{data}}} || \,\mathrm{E}\,\varepsilon_i - \hat{\sigma}_i\,||^2 \rightarrow \min_{\mathrm{E}} \qquad \ldots \text{ estimate model parameter E.} \tag{1}$$

When phrased as least square's problems, provided linear regression problems have a *convex* objective function with a *unique global minimum*. For our example (1), we can find it by evaluating the vanishing derivative, $\partial L/\partial \mathrm{E} = \sum_{i=1}^{n_{\text{data}}} 2\varepsilon_i\,(\mathrm{E}\,\varepsilon_i - \hat{\sigma}_i) \doteq 0$. Here, the minimization problem is not only linear in the model parameter E, but also in the dependent variable ε, and we obtain an explicit solution for the elastic modulus, $\mathrm{E} = \sum_i^{n_{\text{data}}} \varepsilon_i\,\hat{\sigma}_i \,/\, \sum_i^{n_{\text{data}}} \varepsilon_i\,\varepsilon_i$. For linear regression with multiple model parameters $\boldsymbol{\theta}$, for which the minimization problem is a linear function in the dependent variables ε, we obtain a *coupled system of equations*, $\partial L/\partial \boldsymbol{\theta} \doteq \mathbf{0}$, with an explicit solution for the parameter vector $\boldsymbol{\theta}$. For linear regression with one or multiple model parameters $\boldsymbol{\theta}$, for which the minimization problem is a nonlinear function in the dependent variables ε, we obtain a similar set of one or more equations $\partial L/\partial \boldsymbol{\theta} \doteq \mathbf{0}$, which may require an iterative solution for the parameter vector $\boldsymbol{\theta}$. Importantly, *any* regression that is linear in the model parameters $\boldsymbol{\theta}$—independent of whether it is linear, polynomial, or generally nonlinear in the independent variables ε_i—is considered a linear regression problem that, when phrased as a least square's problem with appropriate data, results in a convex quadratic function in the model parameters $\boldsymbol{\theta}$ with a unique global minimum.

Nonlinear regression. Nonlinear regression[44] seeks to model the relationship between a *dependent variable*, in our case the Piola stress $\boldsymbol{P}$, and a set of one or more *independent variables*, in our case the set of deformation gradients $\boldsymbol{F}_i$ at different load levels i, using a function that depends *nonlinearly* on a set of *model parameters* $\boldsymbol{\theta}$.[45] The regression estimates these parameters by minimizing the difference between the predicted stress, $\boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta})$, for given deformation gradients $\boldsymbol{F}_i$ and model parameters $\boldsymbol{\theta}$, and the experimentally measured stresses $\hat{\boldsymbol{P}}_i$, divided by the number of data points n_{data}. Similar to linear regression, we can measure for this difference as the mean squared error based on the L_2-norm,[34] $||(\circ)|| = ||(\circ)||_2 = |(\circ)^2|^{1/2}$, for which the minimization problem becomes

$$L(\boldsymbol{\theta};\boldsymbol{F}) = \frac{1}{n_{\text{data}}}\sum_{i=1}^{n_{\text{data}}} || \,\boldsymbol{P}(\boldsymbol{F}_i,\boldsymbol{\theta}) - \hat{\boldsymbol{P}}_i\,||^2 \rightarrow \min_{\boldsymbol{\theta}} \qquad \ldots \text{ estimate model parameters } \boldsymbol{\theta}. \tag{2}$$

In general, nonlinear regression problems have a *non-convex* objective function with *multiple local minima*. Solving non-convex optimization problems requires iterative algorithms that are at risk of converging to a local minimum instead of the global minimum, and their solution is often highly sensitive to the initial conditions that we select for the parameter vector. Depending on the nature of the problem, the solution we find may involve a large and dense parameter vector $\boldsymbol{\theta}$, and overfitting may occur when the number of parameters is larger than the number of data points, $n_{\text{para}} > n_{\text{data}}$. Notably, even for many data points, we may face overfitting when the data are noisy or not rich enough to sufficiently activate all the parameters. For example, with tension and compression tests alone, we cannot estimate model parameters for shear.

Sparse regression. Sparse regression is a special type of regression that seeks to prevent overfitting by inducing sparsity in the parameter vector $\boldsymbol{\theta}$ by setting a large number of parameters to zero.[33] Sparse regression is particularly useful in high-dimensional settings, since it generates models with a small subset of non-zero parameters,[34] which tends to make the model more interpretable.[35] Historically, the need for sparse regression emerged prominently with the advent of high-dimensional datasets for which the number of parameters can easily exceed the number of independent observations.[46] A prominent example is SINDy, an algorithm for sparse identification in nonlinear dynamics that promotes sparsity through sequential thresholded least-squares by iterating between a partial least-squares fit and a thresholding step to sequentially drop the least relevant terms of a model.[20] Importantly, while these sparsification algorithms converge well in linear regression associated with convex objective functions,[47] their convergence is no longer guaranteed in nonlinear regression with non-convex objective functions. The advantages of sparse regression are *improved interpretability* by reducing the parameter set to only a few non-zero terms; *feature selection* by identifying the most relevant terms; and *reduced risk of overfitting* by promoting simpler models. These advantages come at a price: the disadvantages of sparse regression are *selection bias* by enforcing sparsity of the parameter estimates; *additional hyperparameters* that need to be tuned and require additional attention; and *risk of misspecification* by excluding relevant parameters if sparsity is enforced

too aggressively. In conclusion, sparse regression offers a powerful toolset for high-dimensional modeling, but introduces a trade-off between interpretability and prediction accuracy.

Neural networks. Neural networks are a class of models and algorithms that can approximate a wide range of functions.[48] Their versatility not only makes them a powerful tool for classification, reinforcement learning, and generative tasks, but also for *regression* problems,[49] in our context, for regression in constitutive modeling.[16] Neural networks consist of input, hidden, and output layers with several nodes in each layer. Their parameters are the network weights $\boldsymbol{\theta} = \{w_{i,j}\}$, where $i = 1, \dots, n_{\text{lay}}$ is the number of hidden layers and $j = 1, \dots, n_{\text{nod}}$ is the number of nodes per layer. During training, neural networks effectively perform a regression as they learn their parameters by minimizing a loss function L that penalizes the error between model and data. Similar to the classical nonlinear regression in Equation (2), we can characterize this error as the mean squared error, the L_2-norm of the difference between the stress predicted by the model $\boldsymbol{P}(\boldsymbol{F}_i)$ and the experimentally measured stress $\hat{\boldsymbol{P}}_i$, divided by the number of data points n_{data} to train the model,

$$L(\boldsymbol{\theta}; \boldsymbol{F}) = \frac{1}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} || \boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta}) - \hat{\boldsymbol{P}}_i ||^2 \rightarrow \min \qquad \dots \text{ learn network weights } \boldsymbol{\theta} = \{w_{i,j}\}. \tag{3}$$

However, in contrast to traditional regression tools that have a fixed functional form, neural networks can easily adapt their shape which allows them to model complex functions,[5] either linear[50] or nonlinear[29] in the model parameters $\boldsymbol{\theta}$. The advantages of neural networks are their *universal approximation* that allows them to approximate *any* continuous function for a sufficiently large number of weights,[48] and their inherent *flexibility* that allows them to model high-dimensional nonlinear relationships like the constitutive behavior we seek to model here. Their disadvantages are their *computational complexity*, especially for densely connected architectures with multiple hidden layers; *risk of overfitting* sparse or noisy data; and *lack of interpretability* that generally worsens with the number of layers and makes plain neural networks unsuitable for model discovery tasks.

Sparse neural networks. Sparse neural networks use a special type of network architecture for which a large number of weights are zero. This reduces the number of active connections between the nodes of consecutive layers. Sparsity can be induced *during* training by using special algorithms, or *after* training by pruning.[51] The concepts of sparse neural networks and weight or node pruning—inspired by brain development and synaptic pruning—have gained increasing attention with the rise of deep learning and the need for computational efficiency.[52] The advantages of sparse neural networks are their *computational efficiency* and faster inference times; *reduced risk of overfitting* by promoting smaller model sizes; and their potential for *regularization* and improved *generalization*. The disadvantages are *increased training complexity* to induce sparsity; and *risk of decreased performance* through overly aggressive sparsification or pruning.

The objective of this review is to explore neural networks for which sparsity is induced by a hybrid approach of two combined strategies: *regularization* and *physical constraints*. Towards this goal, first, we review popular subset selection and shrinkage methods to induce sparsity within the general framework of L_p *regularization* in Section 2. Then, we discuss how to leverage physical constraints to induce sparsity within the framework of *constitutive neural networks* in Section 3. Finally, we propose a hybrid approach of L_p regularized constitutive neural networks for automated model discovery and discuss the interpretability and prediction accuracy of their discovered models by means of a library of illustrative examples in Section 4. We conclude by comparing the different approaches and providing guidelines and recommendations in Section 5.

2 | L_p REGULARIZATION

The concept of L_p regularization or bridge regression was first introduced three decades ago in the context of chemometrics with the goal to shrink the parameter space in chemical data analysis.[30] The method has re-gained attention as a powerful tool to promote sparsity in system identification,[20] and, most recently, in discovering constitutive models from data.[25] L_p regularization is a generalized regularization technique that uses the L_p norm of the parameter vector $\boldsymbol{\theta}$, the sum of the pth power of the norm of its $i = 1, \dots, n_{\text{para}}$ coefficients w_i, raised to the inverse power, $|| \boldsymbol{\theta} ||_p = [\sum_{i=1}^{n_{\text{para}}} | w_i |^p]^{1/p}$. Bridge regression constrains the regression (2) by constraining this L_p norm to be smaller than a non-negative

constant $\epsilon \geq 0$,

$$L(\boldsymbol{\theta}; \boldsymbol{F}) = \frac{1}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} || \boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta}) - \hat{\boldsymbol{P}}_i ||^2 \rightarrow \min \qquad \text{subject to} \qquad || \boldsymbol{\theta} ||_p^p = \sum_{i=1}^{n_{\text{para}}} | w_i |^p \leq \epsilon. \tag{4}$$

It proves convenient to reformulate this *constrained regression* problem as a *penalized regression* problem, by penalizing the regression (2) with a penalty term that consists of the L_p norm $|| \boldsymbol{\theta} ||_p^p$, multiplied by a non-negative penalty parameter $\alpha \geq 0$,

$$L(\boldsymbol{\theta}; \boldsymbol{F}) = \frac{1}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} || \boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta}) - \hat{\boldsymbol{P}}_i ||^2 + \alpha || \boldsymbol{\theta} ||_p^p \rightarrow \min \qquad \text{with} \qquad || \boldsymbol{\theta} ||_p^p = \sum_{i=1}^{n_{\text{para}}} | w_i |^p. \tag{5}$$

The constrained and penalized regression problems (4) and (5) are equivalent, which implies that for a given $\epsilon \geq 0$, there exists an $\alpha \geq 0$ such that the two problems share the same solution $\boldsymbol{\theta}$.[53] The flexible power p not only allows us to recover classical regularization techniques like L_2 or L_1 regularization as special cases, but also to interpolate smoothly between these different methods.[30] The advantages of L_p regularization are its inherent *flexibility* by allowing for a continuum of popular regularization techniques for varying powers p; and its potential to effectively *promote sparsity*. Its disadvantages are the added complexity associated with the *selection of hyperparameters*, specifically the penalty parameter α and the power p; and the potential *computational challenges* associated with specific choices for p. Figure 1 illustrates the contours of the regularization term, for varying powers p as $p = [0.25, 0.5, 0.75, 1, 1.5, 2, 4, 8]$, evaluated for two parameters, w_1 and w_2. The top row illustrates the effect of powers smaller than or equal to one, $p \leq 1$; the bottom row illustrates the effect powers larger than one, $p > 1$. In the following, we highlight the most popular special cases of L_p regularization, their history, and their advantages and disadvantages.

L_2 regularization or ridge regression. L_2 regularization, commonly known to as ridge regression, was introduced more than half a century ago to address multicollinearity in regression analysis,[31] and has gained attention for its ability to stabilize parameter estimates, especially when the parameters are closely correlated.[34] It uses the L_2 norm of a vector, the Euclidian norm, the sum of the vector components squared, $|| \boldsymbol{\theta} ||_2 = [\sum_{i=1}^{n_{\text{para}}} | w_i |^2]^{1/2}$. Notably, the L_2 norm does *not* weigh all entries of the vector equally. Instead, it squares the vector entries which makes it highly sensitive to outliers as it penalizes the squared magnitude of the individual parameters w_i. Ridge regression supplements the regression (2) with a penalty term that consists of the L_2 norm multiplied by a penalty parameter α,

$$L(\boldsymbol{\theta}; \boldsymbol{F}) = \frac{1}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} || \boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta}) - \hat{\boldsymbol{P}}_i ||^2 + \alpha || \boldsymbol{\theta} ||_2^2 \rightarrow \min \qquad \text{with} \qquad || \boldsymbol{\theta} ||_2^2 = \sum_{i=1}^{n_{\text{para}}} | w_i |^2. \tag{6}$$

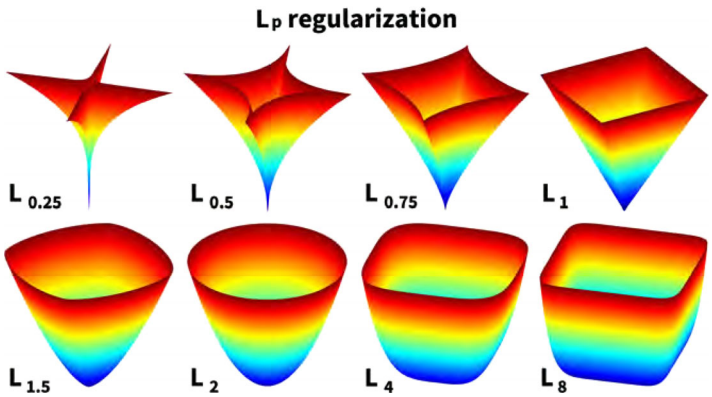

FIGURE 1 ***L_p* regularization.** Contours of regularization term, $L_p = \alpha \sum_{i=1}^{n_{\text{para}}} || \boldsymbol{\theta} ||_p^p$ with $|| \boldsymbol{\theta} ||_p^p = | w_i |^p$, for varying powers, $p = [0.25, 0.5, 0.75, 1, 1.5, 2, 4, 8]$, evaluated for two parameters, w_1 and w_2. For $p \leq 1$, top row, with the special case of L_1 regularization or lasso represented through the pyramid, L_p regularization promotes sparsity by setting some weights exactly to zero, but is no longer strictly convex and can have multiple local minima. For $p > 1$, bottom row, with the special case of L_2 regularization or ridge regression represented through the ellipsoid, L_p regularization promotes stability by reducing outliers, while the regularization term remains convex.

Its advantages are *stability in multicollinearity* by offering stable parameter estimates even in the presence of highly correlated predictors; *managing outliers* and *preventing overfitting* by quadratically penalizing extreme coefficients; and *computational efficiency* even for large datasets. Its disadvantages are *introducing bias*, which may result in underestimating certain coefficients and effects; and its *inability to induce sparsity*, which makes it unsuitable for our current focus of subset selection. Figure 1 shows that, for the special case of L_2 regularization, the regularization term adopts a convex ellipsoidal shape that promotes stability by reducing outliers.

L_1 regularization or lasso. L_1 regularization was initially introduced as a method to analyze seismograms in geophysics almost four decades ago,[32] and has become widely known under the name of lasso, short for least absolute shrinkage and selection operator.[33] Lasso has become popular for producing interpretable models,[4] while exhibiting the same stability properties as ridge regression. It uses the L_1 norm of a vector, the sum of the absolute values of its components, $|| \boldsymbol{\theta} ||_1 = \sum_{i=1}^{n_{\text{para}}} | w_i |$. Because of its similarities with a distance between city blocks, the L_1 norm is often referred to as the Manhattan distance or taxicab norm. Notably, the L_1 norm weighs all entries of the vector *equally* and is therefore less sensitive to outliers than the L_2 norm. Lasso supplements the regression (2) with a penalty term that consists of the L_1 norm multiplied by a penalty parameter α,

$$L(\boldsymbol{\theta}; \boldsymbol{F}) = \frac{1}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} || \boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta}) - \hat{\boldsymbol{P}}_i ||^2 + \alpha || \boldsymbol{\theta} ||_1 \rightarrow \min \qquad \text{with} \qquad || \boldsymbol{\theta} ||_1 = \sum_{i=1}^{n_{\text{para}}} | w_i |. \tag{7}$$

Its advantages are *enabling feature selection* and *inducing sparsity* by reducing some weights exactly to zero which effectively reduces model complexity and improves interpretability; *mitigating overfitting* by constraining the magnitude of the weights, which is especially important when data are limited or high-dimensional; and *providing predictive insights* by identifying the most relevant weights.[34] Its disadvantages are *introducing bias*, which may result in underestimating certain coefficients and effects; and *focusing on selective effects* while discarding others, especially in nuanced multi-effect situations when the weights are closely correlated. Figure 1 shows that, for the special case of L_1 regularization, the regularization term adopts a non-strictly-convex pyramid shape that promotes sparsity by reducing some weights exactly to zero.

$L_{1/2}$ regularization or elastic net. $L_{1/2}$ regularization, also known as elastic net, is a hybrid approach that seeks to combine the benefits of both L_1 and L_2 regularization.[54] The elastic net supplements the regression (2) with two penalty terms in terms of the L_2 and L_1 norms multiplied by two independent penalty parameters α_2 and α_1,

$$L(\boldsymbol{\theta}; \boldsymbol{F}) = \frac{1}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} || \boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta}) - \hat{\boldsymbol{P}}_i ||^2 + \alpha_2 || \boldsymbol{\theta} ||_2^2 + \alpha_1 || \boldsymbol{\theta} ||_1 \rightarrow \min \text{ with } || \boldsymbol{\theta} ||_2^2 = \sum_{i=1}^{n_{\text{para}}} | w_i |^2 \text{ and } || \boldsymbol{\theta} ||_1 = \sum_{i=1}^{n_{\text{para}}} | w_i |. \tag{8}$$

For $\alpha_1 = 0$ and $\alpha_2 = 0$, it recovers the classical ridge regression and lasso as special cases. For $\alpha_1 > 0$ and $\alpha_2 > 0$, $L_{1/2}$ regularization shares many features with L_p regularization with $1 < p < 2$ and generates contours similar, but *not identical* to Figure 1, bottom left. However, in contrast to L_p regularization with $1 < p < 2$,[30] $L_{1/2}$ regularization not only *promotes stability*, but also *induces sparsity* while remaining convex.[34] Its disadvantage is its added *computational complexity*. Since $L_{1/2}$ regularization is *not* a special case of the L_p regularization family, we will not consider it further throughout this study.

$L_{0.5}$ regularization. L_p regularization with powers $0 < p < 1$ has become a popular tool in subset selection since it promotes sparsity more aggressively than simple L_1 regularization. While L_p norms are traditionally defined for powers larger than one, $p \geq 1$, the concept of applying powers smaller than one, $0 < p < 1$, was introduced more than three decades ago in sparse regression of large systems.[30] Notably, for powers smaller than one, the penalty term becomes non-convex and is no longer a norm in the classical sense. For the special case of $p = 0.5$, the penalty term uses the sum of the square roots of the vector components, $|| \boldsymbol{\theta} ||_{0.5} = [\sum_{i=1}^{n_{\text{para}}} \sqrt{|w_i|}]^2$, and we can easily see that this construct no longer satisfies the triangle inequality. $L_{0.5}$ regularization supplements the regression (2) with a penalty term that consists of this term, multiplied by a penalty parameter α,

$$L(\boldsymbol{\theta}; \boldsymbol{F}) = \frac{1}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} || \boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta}) - \hat{\boldsymbol{P}}_i ||^2 + \alpha || \boldsymbol{\theta} ||_{0.5}^{0.5} \rightarrow \min \qquad \text{with} \qquad || \boldsymbol{\theta} ||_{0.5}^{0.5} = \sum_{i=1}^{n_{\text{para}}} \sqrt{|w_i|}. \tag{9}$$

The advantages of $L_{0.5}$ regularization, or more generally of L_p regularization with powers smaller than one, are *enhanced sparsity* potentially leading to more parsimonious models; and *subset selection* especially in high-dimensional datasets.[55] Its disadvantages are its *computational complexity* induced by its non-convex nature; and its *multiple local minima* making subset selection complex and non-unique. Figure 1 shows that, for the special cases of $L_{0.75}$, $L_{0.5}$, and $L_{0.25}$ regularization, the regularization term adopts an increasingly non-convex shape and promotes sparsity by more aggressively setting weights equal to zero.

L_0 regularization or subset selection. L_0 regularization, is a form of subset selection that imposes a penalty on the number of non-zero parameters in a regression model. The origins of selecting a subset of relevant parameters date back to early efforts in regression modeling with the objective to discover parsimonious models with enhanced interpretation and prediction. However, formalizing this idea as an L_0 penalty method and connecting it rigorously to regularization has emerged more prominently with the advent of high-dimensional datasets.[30] The L_0 norm is commonly referred to as sparse norm and is not a norm in a strict mathematical sense. It refers to the pseudo-norm, $||\boldsymbol{\theta}||_0 = \sum_{i=1}^{n_{\text{para}}} I(w_i \neq 0)$, where $I(\circ)$ is the indicator function that is one if the condition inside the parenthesis is true and zero otherwise. As such, the L_0 norm counts the number of non-zero entries in a vector, which implies that this approach directly penalizes model complexity in terms of predictor inclusion. Notably, the L_0 norm is an *explicit*, *discrete* measure of sparsity. It is robust to outliers since it only counts the number of non-zero elements in the parameter vector and does not express preference for smaller or larger entries. L_0 regularization supplements the regression (2) with a penalty term that consists of the L_0 norm, multiplied by a penalty parameter α,

$$L(\boldsymbol{\theta};\boldsymbol{F}) = \frac{1}{n_{\text{data}}}\sum_{i=1}^{n_{\text{data}}} ||\ \boldsymbol{P}(\boldsymbol{F}_i,\boldsymbol{\theta}) - \hat{\boldsymbol{P}}_i\ ||^2 + \alpha\ ||\ \boldsymbol{\theta}\ ||_0 \rightarrow \min \qquad \text{with} \qquad ||\ \boldsymbol{\theta}\ ||_0 = \sum_{i=1}^{n_{\text{para}}} I(w_i \neq 0). \tag{10}$$

Its advantages are its *conceptual simplicity* by providing a direct mechanism for subset selection that directly penalizes non-zero parameters; *reduced overfitting* by promoting fewer non-zero parameters, in particular when data are limited; and *enhanced model interpretability* by focusing only on the relevant terms. Its disadvantages are its *computational complexity* that results from turning continuous model selection into an *NP hard discrete combinatorial problem* with 2^n possible parameter combinations, making it computationally intractable for problems with large parameter sets; its *non-convexity* induced by the L_0 penalty term that leads to optimization challenges related to several local minima; and increased *instability* by discovering models for which slight changes in the data can result in an entirely different parameter set. In the contour plot of Figure 1, L_0 regularization would correspond to two discrete planes along the two parameter axes.

Predictability and interpretability. L_p regularization is an intricate balance between predictability and interpretability: For powers larger than one, $p > 1$, L_p regularization can improve *predictability*, increase robustness, prevent overfitting, and enhance generalization to new data by penalizing outliers and reducing extreme coefficients. For powers equal to or smaller than one, $p \leq 1$, L_p regularization, can improve *interpretability*, promote simpler models, and identify the most influential predictors by encouraging sparsity and forcing some coefficients exactly to zero. Taken together, L_p regularization is a trade-off between interpretability and predictability, between simplicity and accuracy, and between bias and variance. Two hyperparameters, the power p and the regularization strength α, allow us to fine-tuning of this balance. Throughout this manuscript, we will provide a library of systematic examples that illustrate the sensitivity of L_p regularization with respect to these two hyperparameters.

3 | NEURAL NETWORKS

In this study, we adopt the concept of neural networks to perform regression in constitutive modeling with the objective to improve both *predictability* and *interpretability*. To demonstrate that our strategy generalizes to different types of neural networks, we compare two constitutive neural networks that have recently become popular in the context of automated model discovery.[29,50] Both networks are *sparse neural networks* by design, where sparsity is inspired by the underlying physics of hyperelasticity. In their input layer, they use characteristic features of the deformation gradient $\boldsymbol{F}$ to a priori satisfy the kinematic constraint of *material objectivity*, and acknowledge a characteristic isotropic and incompressible material behavior by satisfying the constraints of *material symmetry* and *incompressibility*.[15] In their output layer, they

learn a free energy function ψ from which they derive the nominal stress $\boldsymbol{P}$ to a priori satisfy the dissipation inequality and, with it, *thermodynamic consistency*.[10] In their hidden layers, both networks use a special set of custom-designed activation functions to a priori satisfy *physical constraints*[3] and a particular network architecture to satisfy *polyconvexity*.[9,12]

Invariant based and principal stretch based neural networks. We explore two types of neural networks that use different types of activation functions to represent two different classes of constitutive models: invariant and principal stretch based.[56,57] Both networks are *generalizations* of popular constitutive models that include widely used hyperelastic models as special cases.[58–61] They are *interpretable* by design and their weights translate into physically meaningful parameters with physical units and a physical interpretation.[17] Yet, there is a major difference between both models: the invariant model uses *different functional forms* for each activation function and results in a *nonlinear regression problem*, whereas the principal stretch model uses the *same functional form with different but fixed exponents* and results in a *linear regression problem*. This has critical implications on the convexity of the objective function, and with it, on the nature of the solution.

Data. We train our networks on both synthetic and real data from tension, compression, and shear tests. For the *synthetic data*, we generate stretch stress pairs for fixed parameters through forward simulation. Specifically, we calculate stresses over a wide range of tensile stretches, compressive stretches, and shear strains in ten equidistant increments, resulting in three data sets with eleven stretch-stress pairs each. For the *real data*, we extract stretch stress pairs from our previously published human brain experiments on 5 mm gray matter tissue cubes.[62,63] Specifically, we use the reported stresses averaged over $n = 15$ tensile, $n = 17$ compressive, and $n = 35$ shear experiments, in sixteen equidistant increments, resulting in three data sets with seventeen stretch-stress pairs each.[29] All data are available on our GitHub repository, https://github.com/LivingMatterLab/CANN.

3.1 | Invariant based neural network

Invariant based constitutive neural networks take the deformation gradient $\boldsymbol{F}$ as input and predict the free energy function ψ as output from which we calculate the stress $\boldsymbol{P} = \partial\psi/\partial\boldsymbol{F}$. From the deformation gradient, they extract a set of invariants, in our example I_1 and I_2, and feed them into its two hidden layers.[29,64] The first layer generates the powers of the invariants, in our example the first and second $(\circ)$ and $(\circ)^2$, and the second layer subjects these powers to specific functions, in our example to the identity and exponential $(\circ)$ and $\exp(\circ)$. The free energy function ψ is a sum of the resulting eight terms. Figure 2 illustrates the invariant based constitutive neural network with the eight functional building blocks highlighted in color, where the hot red colors relate to the first invariant and the cold blue colors to the second. During training, the network *autonomously* discovers the best model, out of $2^8 = 256$ possible combinations of terms, and *simultaneously* learns its model parameters $\boldsymbol{\theta} = \{ w_{i,j} \}$. It minimizes the loss function (3), the difference between the stress predicted by

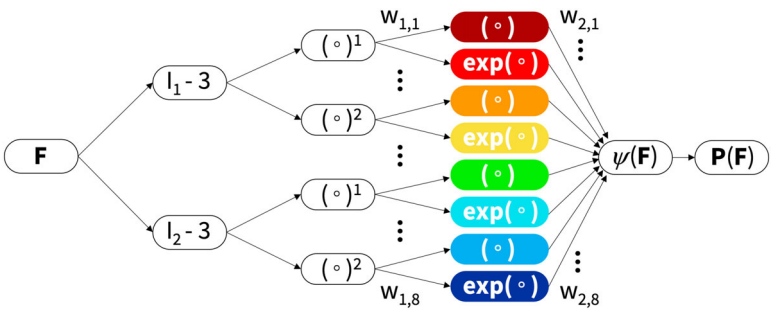

FIGURE 2 **Invariant based neural network for automated model discovery.** The network takes the deformation gradient $\boldsymbol{F}$ as input and outputs the free energy function ψ from which we calculate the stress $\boldsymbol{P} = \partial\psi/\partial\boldsymbol{F}$. The network is invariant based, it first calculates the invariants I_1 and I_2, and feeds them into its two hidden layers. The first layer generates the first and second powers $(\circ)$ and $(\circ)^2$ of the invariants and the second layer applies the identity and exponential function $(\circ)$ and $\exp(\circ)$ to these powers. The free energy function ψ is a function of the eight color-coded terms. During training, the network discovers the best model, of $2^8 = 256$ possible combinations of terms, to explain the experimental data $\hat{\boldsymbol{P}}$.

the model $\boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta})$ and the experimentally measured stress $\hat{\boldsymbol{P}}_i$, divided by the number of data points used for training n_{data},

$$L(\boldsymbol{\theta}; \boldsymbol{F}) = \frac{1}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} || \, \boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta}) - \hat{\boldsymbol{P}}_i \, ||^2 \rightarrow \min . \tag{11}$$

To ensure thermodynamic consistency, the network does not learn the stress directly,[16] but rather derives it from the free energy function.[15] For our example in Figure 2, free energy function takes the following explicit representation,

$$\begin{aligned} \psi(I_1, I_2) = & \; w_{2,1} w_{1,1} \, [I_1 - 3] \; + w_{2,2} \, [\exp(w_{1,2} \, [I_1 - 3] \,) - 1] \\ & + w_{2,3} w_{1,3} \, [I_1 - 3]^2 + w_{2,4} \, [\exp(w_{1,4} \, [I_1 - 3]^2) - 1] \\ & + w_{2,5} w_{1,5} \, [I_2 - 3] \; + w_{2,6} \, [\exp(w_{1,6} \, [I_2 - 3] \,) - 1] \\ & + w_{2,7} w_{1,7} \, [I_2 - 3]^2 + w_{2,8} \, [\exp(w_{1,8} \, [I_2 - 3]^2) - 1], \end{aligned} \tag{12}$$

with the following derivatives with respect to the invariants I_1 and I_2,

$$\begin{aligned} \frac{\partial \psi}{\partial I_1} &= w_{2,1} w_{1,1} + w_{2,2} w_{1,2} \exp(\, w_{1,2} [I_1 - 3]) + 2 \, [I_1 - 3] [w_{2,3} w_{1,3} + w_{2,4} w_{1,4} \exp(\, w_{1,4} [I_1 - 3]^2)] \\ \frac{\partial \psi}{\partial I_2} &= w_{2,5} w_{1,5} + w_{2,6} w_{1,6} \exp(\, w_{1,6} [I_2 - 3]) + 2 \, [I_2 - 3] [w_{2,7} w_{1,7} + w_{2,8} w_{1,8} \exp(\, w_{1,8} [I_2 - 3]^2)]. \end{aligned} \tag{13}$$

Using the second law of thermodynamics, we can derive the Piola stress, $\boldsymbol{P} = \partial \psi / \partial \boldsymbol{F}$, as thermodynamically conjugate to the deformation gradient $\boldsymbol{F}$,[56]

$$\boldsymbol{P} = \frac{\partial \psi}{\partial I_1} \cdot \frac{\partial I_1}{\partial \boldsymbol{F}} + \frac{\partial \psi}{\partial I_2} \cdot \frac{\partial I_2}{\partial \boldsymbol{F}} - p \, \boldsymbol{F}^{-\text{t}}, \tag{14}$$

where the term $p \, \boldsymbol{F}^{-\text{t}}$ ensures perfect incompressibility in terms of the pressure p that we determine from the boundary conditions. For the network free energy (12), the Piola stress is

$$\begin{aligned} \boldsymbol{P} \; = \; & [\qquad\quad w_{2,1} w_{1,1} \; + w_{2,2} w_{1,2} \;\; \exp(w_{1,2} \;\; [I_1 - 3] \;) \\ & + \; 2 \, [\, I_1 - 3 \,] [w_{2,3} w_{1,3} \; + w_{2,4} w_{1,4} \;\; \exp(w_{1,4} \;\; [I_1 - 3]^2)] \, \partial I_1 / \partial \boldsymbol{F} \\ & + \; [\qquad\quad w_{2,5} w_{1,5} \; + w_{2,6} w_{1,6} \;\; \exp(w_{1,6} \;\; [I_2 - 3] \;) \\ & + \; 2 \, [\, I_2 - 3 \,] [w_{2,7} w_{1,7} \; + w_{2,8} w_{1,8} \;\; \exp(w_{1,8} \;\; [I_2 - 3]^2)] \, \partial I_2 / \partial \boldsymbol{F} - p \, \boldsymbol{F}^{-\text{t}} \, . \end{aligned} \tag{15}$$

Notably, the Piola stress of the invariant based network (15) is a nonlinear function in the network weights $w_{i,j}$, which translates the loss function (11) into a *nonlinear regression* problem, with possibly multiple local minima. For this particular format, one of the first two weights of each row becomes redundant, and we can reduce the set of network parameters to twelve, $\boldsymbol{\theta} = [\, (w_{1,1} w_{2,1}), w_{1,2}, w_{2,2}, (w_{1,3} w_{2,3}), w_{1,4}, w_{2,4} (w_{1,5} w_{2,5}), w_{1,6}, w_{2,6}, (w_{1,7} w_{2,7}), w_{1,8}, w_{2,8} \,]$. We train our invariant based network with tension, compression, and shear data and rewrite the loss function (11) in terms of two contributions that minimize the error between the normal and shear stresses predicted by the model, $P_{11}(\lambda_i)$ and $P_{12}(\gamma_i)$, and the data, $\hat{P}_{11,i}$ and $\hat{P}_{12,i}$, where n_{11} and n_{12} denote the different stretch and shear levels λ and γ,

$$L(\boldsymbol{\theta}; \lambda, \gamma) = \frac{1}{n_{11}} \sum_{i=1}^{n_{11}} || \, P_{11}(\lambda_i) - \hat{P}_{11,i} \, ||^2 + \frac{1}{n_{12}} \sum_{i=1}^{n_{12}} || \, P_{12}(\gamma_i) - \hat{P}_{12,i} \, ||^2 \rightarrow \min . \tag{16}$$

To explore the effect of *scaling* of the three individual stress terms, alternatively, we weigh all three experiments equally, and also train the network by minimizing the error between the normalized tensile, compressive, and shear stresses predicted by the model $P_{\text{ten}}(\lambda_i)$, $P_{\text{com}}(\lambda_i)$, and $P_{\text{shr}}(\gamma_i)$, and the data $\hat{P}_{\text{ten},i}$, $\hat{P}_{\text{com},i}$, and $\hat{P}_{\text{shr},i}$, normalized by the maximum recorded tensile, compressive, and shear stresses, $\hat{P}_{\text{ten}}^{\text{max}}$, $\hat{P}_{\text{com}}^{\text{min}}$, and $\hat{P}_{\text{shr}}^{\text{max}}$, where n_{ten}, n_{com}, and n_{shr} denote the different

stretch and shear levels λ and γ,

$$L(\boldsymbol{\theta};\lambda,\gamma)=\frac{1}{n_{\text{ten}}}\sum_{i=1}^{n_{\text{ten}}}\left|\left|\frac{P_{\text{ten}}(\lambda_i)-\hat{P}_{\text{ten},i}}{\hat{P}^{\text{max}}_{\text{ten}}}\right|\right|^2+\frac{1}{n_{\text{com}}}\sum_{i=1}^{n_{\text{com}}}\left|\left|\frac{P_{\text{com}}(\lambda_i)-\hat{P}_{\text{com},i}}{\hat{P}^{\text{min}}_{\text{com}}}\right|\right|^2+\frac{1}{n_{\text{shr}}}\sum_{i=1}^{n_{\text{shr}}}\left|\left|\frac{P_{\text{shr}}(\gamma_i)-\hat{P}_{\text{shr},i}}{\hat{P}^{\text{max}}_{\text{shr}}}\right|\right|^2\rightarrow\min. \tag{17}$$

Below, we briefly derive the explicit analytical expressions for the Piola stresses $P_{11}(\lambda)$ in uniaxial tension and compression and $P_{12}(\gamma)$ in simple shear, such that the tensile stress is $P_{\text{ten}}=P_{11}$ for $\lambda>1$, the compressive stress is $P_{\text{com}}=P_{11}$ for $\lambda<1$, and the shear stress is $P_{\text{shr}}=P_{12}$ for all γ.

Uniaxial tension and compression. For the special case of uniaxial tension and compression in terms of the stretch λ, with $\lambda_1=\lambda$ and $\lambda_2=\lambda^{-1/2}$ and $\lambda_3=\lambda^{-1/2}$, the invariants take the following form,

$$I_1=\lambda^2+\frac{2}{\lambda}\quad\text{and}\quad I_2=2\lambda+\frac{1}{\lambda^2}\quad\text{and}\quad I_3=1. \tag{18}$$

Using Equation (14) and the zero normal stress condition, $P_{22}=P_{33}=0$, we obtain the following expression for the uniaxial stress stretch relation,

$$P_{11}=2\left[\frac{\partial\psi}{\partial I_1}+\frac{1}{\lambda}\frac{\partial\psi}{\partial I_2}\right]\left[\lambda-\frac{1}{\lambda^2}\right], \tag{19}$$

which translates into the following explicit expression between our network stress P_{11} and the uniaxial stretch λ,

$$\begin{aligned}P_{11}=&\,2[[w_{2,1}w_{1,1}+w_{2,2}w_{1,2}\exp(w_{1,2}[I_1-3])+2\,[I_1-3][w_{2,3}w_{1,3}+w_{2,4}w_{1,4}\exp(w_{1,4}[I_1-3]^2)]\\&+\frac{1}{\lambda}\,[w_{2,5}w_{1,5}+w_{2,6}w_{1,6}\exp(w_{1,6}[I_2-3])+2\,[I_2-3][w_{2,7}w_{1,7}+w_{2,8}w_{1,8}\exp(w_{1,8}[I_2-3]^2)]]\left[\lambda-\frac{1}{\lambda^2}\right].\end{aligned} \tag{20}$$

Simple shear. For the special case of simple shear, in terms of the shear γ, with $\lambda_1=\frac{1}{2}\,[\,+\gamma+\sqrt{4+\gamma^2}\,]$ and $\lambda_2=\frac{1}{2}\,[\,-\gamma+\sqrt{4+\gamma^2}\,]$ and $\lambda_3=1$, the invariants take the following form,

$$I_1=3+\gamma^2\quad\text{and}\quad I_2=3+\gamma^2\quad\text{and}\quad I_3=1. \tag{21}$$

Using Equation (14), we obtain the following shear stress stretch relation,

$$P_{12}=2\left[\frac{\partial\psi}{\partial I_1}+\frac{\partial\psi}{\partial I_2}\right]\gamma, \tag{22}$$

which translates into the following explicit expression between our network shear stress P_{12} and the shear strain γ,

$$\begin{aligned}P_{12}=&\,2\,[w_{2,1}w_{1,1}+w_{2,2}w_{1,2}\exp(\,w_{1,2}[I_1-3])+2\,[I_1-3][w_{2,3}w_{1,3}+w_{2,4}w_{1,4}\exp(\,w_{1,4}[I_1-3]^2)\\&+w_{2,5}w_{1,5}+w_{2,6}w_{1,6}\exp(\,w_{1,6}[I_2-3])+2\,[I_2-3][w_{2,7}w_{1,7}+w_{2,8}w_{1,8}\exp(\,w_{1,8}[I_2-3]^2)]\,\gamma\,.\end{aligned} \tag{23}$$

Figure 3 illustrates the contours of the loss function $L(\boldsymbol{\theta};\lambda,\gamma)$ for all possible two-term models of the invariant based network in Figure 2. By combining any two terms of the model and setting all other weights equal to zero, we can generate 28 possible models. For these 28 combinations of two terms, we evaluate two versions of the loss function, non-normalized from Equation (16) and normalized from Equation (17), using the invariant based definitions of the normal stress (20) and shear stress (23). First, we generate synthetic data, $\hat{P}_{11,i}$ and $\hat{P}_{12,i}$, for tensile stretches of $\lambda=[1.0,\,\ldots\,,2.0]$, compressive stretches of $\lambda=[1.0,\,\ldots\,,0.5]$, and shear strains of $\gamma=[0.0,\,\ldots\,,0.5]$, in ten equidistant increments each, assuming an exact solution with fixed weights of the first layer, $w_{1,1}=w_{1,3}=w_{1,5}=w_{1,7}=1.0$ and $w_{1,2}=w_{1,4}=w_{1,6}=w_{1,8}=0.25$, and a pair of non-zero weights of the second layer, $w_{2,i}=1$ and $w_{2,j}=1$, while fixing the remaining six weights of the second layer equal to zero. This results in 28 training data sets of eleven stretch-stress pairs each, for tension, compression, and shear. Second, we vary the two non-zero network weights in the ranges $w_i=[0,\,\ldots\,,2]$ and $w_j=[0,\,\ldots\,,2]$. For each pair of weights, we evaluate the normal and shear stresses $P_{11}(\lambda_i)$ and $P_{12}(\gamma_i)$ using Equations (20) and (23), and extract

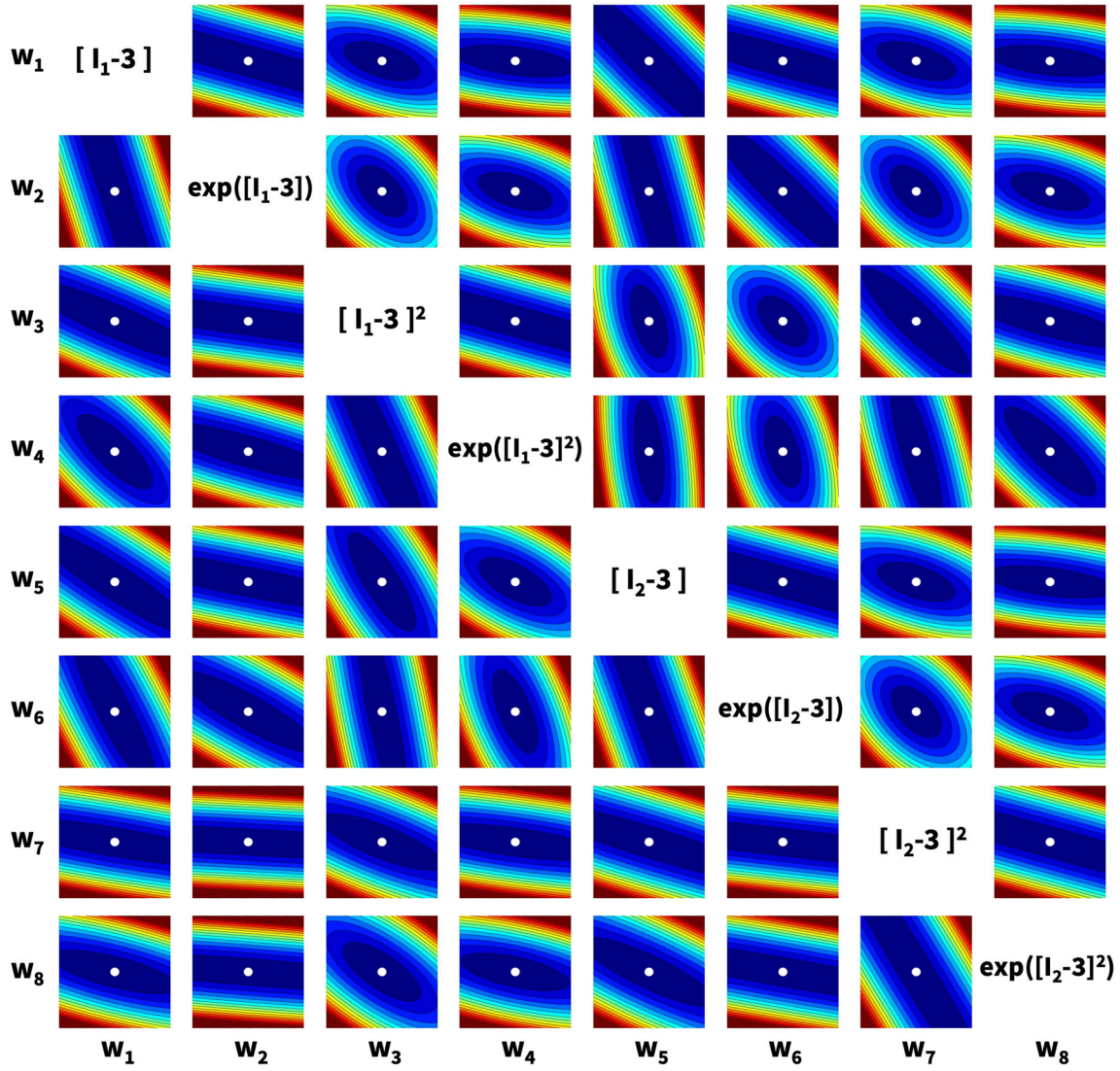

FIGURE 3 **Loss functions of invariant based neural network.** Contours of the loss function $L(\boldsymbol{\theta}; \lambda, \gamma)$ for all 28 possible two-term models of the invariant based constitutive neural network in Figure 2. The loss function is evaluated across tensile stretches $\lambda = [1.0, \dots, 2.0]$, compressive stretches $\lambda = [1.0, \dots, 0.5]$, and shear strains $\gamma = [0.0, \dots, 0.5]$, with network weights in the ranges $w_i = [0, \dots, 2]$ and $w_j = [0, \dots, 2]$. The minimum of the loss function indicates the exact solution $w_i = 1$ and $w_j = 1$, represented through the white circle. The lower triangle illustrates the non-normalized loss function (16), the upper triangle illustrates the normalized loss function (17). All loss functions are convex, with contours varying from ellipsoids to valleys with long ridges, highlighting the collinearity of some w_i and w_j pairs.

the tensile, compressive, and shear stresses, $P_{\text{ten}_i}(\lambda)$, $P_{\text{com}_i}(\lambda)$, and $P_{\text{shr}_i}(\gamma)$. Third, we evaluate the *non-normalized* loss function (16) as the mean squared error between the model stresses $P_{11}(\lambda_i)$ and $P_{12}(\gamma_i)$ and the synthetically generated data stresses $\hat{P}_{11,i}$ and $\hat{P}_{12,i}$, and plot its contours for each pair of weights w_i and w_j in the lower triangle. Next, we evaluate the *normalized* loss function (17) as the normalized mean squared error between the model stresses $P_{\text{ten}}(\lambda_i)$, $P_{\text{com}}(\lambda_i)$, and $P_{\text{shr}}(\gamma_i)$, and the synthetically generated data stresses $\hat{P}_{\text{ten},i}$, $\hat{P}_{\text{com},i}$, and $\hat{P}_{\text{shr},i}$, and plot its contours for each pair of weights w_i and w_j in the upper triangle.

Each loss function takes a minimum of $L(\boldsymbol{\theta}; \lambda, \gamma) = 0$ for the *exact solution*, $w_i = 1$ and $w_j = 1$, indicated through the white circles. From this minimum, both versions of the loss function, non-normalized and normalized, increase with both, decreasing and increasing weights w_i and w_j, and remain *convex* within the entire domain, for all 28 two-term models. Yet, the contours of the loss function vary significantly for different pairs of weights w_i and w_j, indicating its sensitivity with respect to the individual terms: some pairs of weights generate loss functions of ellipsoidal shape, for example, the

$\{ w_2, w_3 \}$, $\{ w_2, w_7 \}$, $\{ w_3, w_6 \}$, $\{ w_6, w_7 \}$ pairs in the normalized upper triangle, suggesting that in the studied stretch and shear range, these terms are *non-collinear*, and would represent a rich base for a potential constitutive model. Other pairs of weights generate loss functions with long ridges parallel to the parameter axes, for example, the $\{ w_2, w_7 \}$, $\{ w_2, w_8 \}$, $\{ w_6, w_7 \}$, $\{ w_6, w_8 \}$ pairs in the non-normalized lower triangle, suggesting that these terms are almost *collinear* and not well suited as an independent base for a constitutive model. On average, the normalized pairs in the upper triangle seem to generate more convex loss functions than the non-normalized pairs in the lower triangle, suggesting that normalization helps to generate more convex loss functions, a richer functional base, and a more robust solution overall. Notably, the $\{ w_1, w_5 \}$ model in the first row and fifth column and the $\{ w_5, w_1 \}$ model in the fifth row and first column combine the linear terms in the first and second invariants, $[I_1 - 3]$ and $[I_2 - 3]$, and represent the popular Mooney Rivlin model for rubber-like materials.[59,60] Overall, this simple example only illustrates the 28 two-term models out of a total set of all 256 possible models, only considers a limited stretch and shear range λ and γ, and only screens a narrow window of parameter ranges w_i. Even within these limitations, the contours of loss functions are rather difficult to interpret, making it difficult to comprehend the full potential of the entire network, even though it only consists of eight distinct terms.

3.2 | Principal stretch based neural network

Principal stretch based constitutive neural networks take the deformation gradient $\boldsymbol{F}$ as input and predict the free energy function ψ as output from which we calculate the stress $\boldsymbol{P} = \partial\psi/\partial\boldsymbol{F}$. From the deformation gradient, they extract the principal stretches, λ_1, λ_2, and λ_3, and feed them into the hidden layer.[50,65] The hidden layer applies eight different exponents $(\lambda_1^n + \lambda_2^n + \lambda_3^n - 3)$ to these stretches. The free energy function ψ is a sum of the resulting eight terms. Figure 4 illustrates the principal stretch based constitutive neural network with the eight functional building blocks highlighted in color, where the dark red and green terms are identical to the dark red and green terms of the invariant based network in Figure 2, while the other six terms are different. During training, the network *autonomously* discovers the best model, out of $2^8 = 256$ possible combinations of terms, and *simultaneously* learns its model parameters $\boldsymbol{\theta} = \{ w_i \}$. It minimizes the loss function (3), the difference between the stress predicted by the model $\boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta})$ and the experimentally measured stress $\hat{\boldsymbol{P}}_i$, divided by the number of data points used for training n_{data},

$$L(\boldsymbol{\theta}; \boldsymbol{F}) = \frac{1}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} || \, \boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta}) - \hat{\boldsymbol{P}}_i \, ||^2 \rightarrow \min . \tag{24}$$

The free energy of the principal stretch based model takes the following explicit representation,[57,66] $\psi(\lambda_1, \lambda_2, \lambda_3) = \sum_{i=1}^{n_{\text{term}}} w_i \, [\, \lambda_1^{\alpha_i} + \lambda_2^{\alpha_i} + \lambda_3^{\alpha_i} - 3 \,]$, where the individual weights $w_i = \mu_i/\alpha_i$ correspond to the shear moduli μ_i divided by the exponents α_i. For the eight-term model in Figure 4, we fix these exponents to $\alpha = [+2, +4, +6, +8, -2, -4, -6 - 8 \,]$, such that the free energy becomes a sum of the following $n_{\text{term}} = 8$ terms,

$$\begin{aligned} \psi(\lambda_1, \lambda_2, \lambda_3) = \sum_{i=1}^{3} & w_1 \, [\, \lambda_i^{+2} - 1 \,] + w_2 \, [\, \lambda_i^{+4} - 1 \,] + w_3 \, [\, \lambda_i^{+6} - 1 \,] + w_4 \, [\, \lambda_i^{+8} - 1 \,] \\ & + w_5 \, [\, \lambda_i^{-2} - 1 \,] + w_6 \, [\, \lambda_i^{-4} - 1 \,] + w_7 \, [\, \lambda_i^{-6} - 1 \,] + w_8 \, [\, \lambda_i^{-8} - 1 \,] . \end{aligned} \tag{25}$$

and its derivatives with respect to the principal stretches λ_i takes the following form,

$$\frac{\partial\psi}{\partial\lambda_i} = 2 \, w_1 \lambda_i^{+1} + 4 \, w_2 \lambda_i^{+3} + 6 \, w_3 \lambda_i^{+5} + 8 \, w_4 \lambda_i^{+7} - 2 \, w_5 \lambda_i^{-3} - 4 \, w_6 \lambda_i^{-5} - 6 \, w_7 \lambda_i^{-7} - 8 \, w_8 \lambda_i^{-9} . \tag{26}$$

Using the second law of thermodynamics, we can derive the Piola stress, $\boldsymbol{P} = \partial\psi/\partial\boldsymbol{F}$, as thermodynamically conjugate to the deformation gradient $\boldsymbol{F}$,[57]

$$\boldsymbol{P} = \sum_{i=1}^{3} \frac{\partial\psi}{\lambda_i} \boldsymbol{n}_i \otimes \boldsymbol{N}_i - p \, \boldsymbol{F}^{-\text{t}} , \tag{27}$$

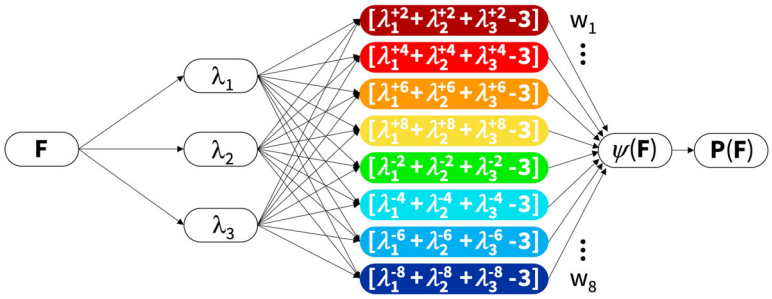

FIGURE 4 **Principal stretch based neural network for automated model discovery.** The network takes the deformation gradient $\boldsymbol{F}$ as input and outputs the free energy function ψ from which we calculate the stress $\boldsymbol{P} = \partial\psi/\partial\boldsymbol{F}$. The network is principal stretch based, it first calculates the principal stretches λ_1 and λ_2 and λ_3, and feeds them into its hidden layer. The hidden layer applies eight different exponents $(\lambda_1^n + \lambda_2^n + \lambda_3^n - 3)$ to these principal stretches. The free energy function ψ is a function of the eight color-coded terms. During training, the network discovers the best model, of $2^8 = 256$ possible combinations of terms, to explain the experimental data $\hat{\boldsymbol{P}}$.

where $\boldsymbol{N}_i$ and $\boldsymbol{n}_i = \boldsymbol{F} \cdot \boldsymbol{N}_i$ are the eigenvectors in the undeformed and deformed configurations, and the term $p\,\boldsymbol{F}^{-\mathrm{t}}$ ensures perfect incompressibilty in terms of the pressure p that we determine from the boundary conditions. For the network free energy (25), the Piola stress is

$$\begin{aligned}\boldsymbol{P} = \sum_{i=1}^{3} [\,2\,w_1\,[\,\lambda_i^{+1} - 1\,] + 4\,w_2\,[\,\lambda_i^{+3} - 1\,] + 6\,w_3\,[\,\lambda_i^{+5} - 1\,] + 8\,w_4\,[\,\lambda_i^{+7} - 1\,] \\ - 2\,w_5\,[\,\lambda_i^{-3} - 1\,] - 4\,w_6\,[\,\lambda_i^{-5} - 1\,] - 6\,w_7\,[\,\lambda_i^{-7} - 1\,] - 8\,w_8\,[\,\lambda_i^{-9} - 1\,]\,]\,\boldsymbol{n}_i \otimes \boldsymbol{N}_i - p\boldsymbol{F}^{-\mathrm{t}},\end{aligned} \tag{28}$$

parameterized in terms of eight network weights, $\boldsymbol{\theta} = [\ w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8\ ]$. Notably, the Piola stress of the principal stretch based network (28) with fixed exponents is a linear function in the network weights w_i, which translates the loss function (24) into a *linear regression* problem, with a single unique global minimum.[67] We train our principal stretch based network with tension, compression, and shear data and rewrite the loss function (24) in terms of two contributions that minimize the error between the tensile and compressive stresses predicted by the model $P_{11}(\lambda_i)$ and the data $\hat{P}_{11,i}$, and between the shear stresses predicted by the model $P_{12}(\gamma_i)$ and the data $\hat{P}_{12,i}$, where we include data from n_{11} different stretch levels λ and n_{12} different shear levels γ,

$$L(\boldsymbol{\theta}; \lambda, \gamma) = \frac{1}{n_{11}} \sum_{i=1}^{n_{11}} || \ P_{11}(\lambda_i) - \hat{P}_{11,i} \ ||^2 + \frac{1}{n_{12}} \sum_{i=1}^{n_{12}} || \ P_{12}(\gamma_i) - \hat{P}_{12,i} \ ||^2 \to \min. \tag{29}$$

For comparison, similar to the invariant based network in the Section 3.1, we also train the network by minimizing the error between the normalized tensile, compressive, and shear stresses predicted by the model $P_{\mathrm{ten}}(\lambda_i)$, $P_{\mathrm{com}}(\lambda_i)$, and $P_{\mathrm{shr}}(\gamma_i)$, and the data $\hat{P}_{\mathrm{ten},i}$, $\hat{P}_{\mathrm{com},i}$, and $\hat{P}_{\mathrm{shr},i}$, normalized by the maximum recorded tensile, compressive, and shear stresses, $\hat{P}_{\mathrm{ten}}^{\mathrm{max}}$, $\hat{P}_{\mathrm{com}}^{\mathrm{min}}$, and $\hat{P}_{\mathrm{shr}}^{\mathrm{max}}$, where n_{ten}, n_{com}, and n_{shr} denote the different stretch and shear levels λ and γ,

$$L(\boldsymbol{\theta}; \lambda, \gamma) = \frac{1}{n_{\mathrm{ten}}} \sum_{i=1}^{n_{\mathrm{ten}}} \left|\left| \frac{P_{\mathrm{ten}}(\lambda_i) - \hat{P}_{\mathrm{ten},i}}{\hat{P}_{\mathrm{ten}}^{\mathrm{max}}} \right|\right|^2 + \frac{1}{n_{\mathrm{com}}} \sum_{i=1}^{n_{\mathrm{com}}} \left|\left| \frac{P_{\mathrm{com}}(\lambda_i) - \hat{P}_{\mathrm{com},i}}{\hat{P}_{\mathrm{com}}^{\mathrm{min}}} \right|\right|^2 + \frac{1}{n_{\mathrm{shr}}} \sum_{i=1}^{n_{\mathrm{shr}}} \left|\left| \frac{P_{\mathrm{shr}}(\gamma_i) - \hat{P}_{\mathrm{shr},i}}{\hat{P}_{\mathrm{shr}}^{\mathrm{max}}} \right|\right|^2 \to \min. \tag{30}$$

Below, we briefly derive the explicit analytical expressions for the Piola stresses $P_{11}(\lambda)$ in uniaxial tension and compression and $P_{12}(\gamma)$ in simple shear, such that the tensile stress is $P_{\mathrm{ten}} = P_{11}$ for $\lambda > 1$, the compressive stress is $P_{\mathrm{com}} = P_{11}$ for $\lambda < 1$, and the shear stress is $P_{\mathrm{shr}} = P_{12}$ for all γ.

Uniaxial tension and compression. For the special case of uniaxial tension and compression in terms of the stretch λ, the principal stretches are

$$\lambda_1 = \lambda \qquad \text{and} \qquad \lambda_2 = \lambda^{-1/2} \qquad \text{and} \qquad \lambda_3 = \lambda^{-1/2}. \tag{31}$$

Using Equation (27) and the zero normal stress condition, $P_{22} = P_{33} = 0$, we obtain the following expression for the uniaxial stress stretch relation,

$$P_{11} = \frac{1}{\lambda} \sum_{i=1}^{n_{\text{term}}} \alpha_i \, w_i \, [\, \lambda^{\alpha_i} - \lambda^{-\alpha_i/2} \,], \tag{32}$$

which translates into the following explicit expression between our network stress P_{11} and the uniaxial stretch λ,

$$\begin{aligned} P_{11} = \frac{1}{\lambda} [\, & 2\, w_1 \, [\, \lambda^{+2} - \lambda^{-1} \,] + 4\, w_2 \, [\, \lambda^{+4} - \lambda^{-2} \,] + 6\, w_3 \, [\, \lambda^{+6} - \lambda^{-3} \,] + 8\, w_4 \, [\, \lambda^{+8} - \lambda^{-4} \,] \\ & - 2\, w_5 \, [\, \lambda^{-2} - \lambda^{+1} \,] - 4\, w_6 \, [\, \lambda^{-4} - \lambda^{+2} \,] - 6\, w_7 \, [\, \lambda^{-6} - \lambda^{+3} \,] - 8\, w_8 \, [\, \lambda^{-8} - \lambda^{+4} \,] \,]. \end{aligned} \tag{33}$$

Simple shear. For the special case of simple shear, in terms of the shear γ, we obtain the principal stretches,

$$\lambda_1 = \frac{\gamma + \sqrt{4 + \gamma^2}}{2} \quad \text{and} \quad \lambda_2 = \frac{-\gamma + \sqrt{4 + \gamma^2}}{2} = \frac{1}{\lambda_1} \quad \text{and} \quad \lambda_3 = 1. \tag{34}$$

Using Equation (27), we obtain the following expression for the shear stress stretch relation,

$$P_{12} = \frac{1}{1 + \lambda^2} \sum_{i=1}^{n_{\text{term}}} \alpha_i \, w_i \, [\, \lambda^{\alpha_i + 1} - \lambda^{1 - \alpha_i} \,] \qquad \text{with} \qquad \lambda = \frac{1}{2} \left[\, \gamma + \sqrt{4 + \gamma^2} \,\right] = \lambda_1 = \frac{1}{\lambda_2}, \tag{35}$$

which translates into the following explicit expression between our network shear stress P_{12} and shear strain γ,

$$\begin{aligned} P_{12} = \frac{1}{1 + \lambda^2} [\, & 2\, w_1 \, [\, \lambda^{+3} - \lambda^{-1} \,] + 4\, w_2 \, [\, \lambda^{+5} - \lambda^{-3} \,] + 6\, w_3 \, [\, \lambda^{+7} - \lambda^{-5} \,] + 8\, w_4 \, [\, \lambda^{+9} - \lambda^{-7} \,] \\ & - 2\, w_5 \, [\, \lambda^{-1} - \lambda^{+3} \,] - 4\, w_6 \, [\, \lambda^{-3} - \lambda^{+5} \,] - 6\, w_7 \, [\, \lambda^{-5} - \lambda^{+7} \,] - 8\, w_8 \, [\, \lambda^{-7} - \lambda^{+9} \,] \,]. \end{aligned} \tag{36}$$

Figure 5 illustrates the contours of the loss function $L(\boldsymbol{\theta}; \lambda, \gamma)$ for all possible two-term models of the principal stretch based network in Figure 4. By combining any two terms of the model and setting all other second-layer weights equal to zero, we can generate 28 possible models. For these 28 combinations of two terms, we evaluate two versions of the loss function, non-normalized from Equation (29) and normalized from Equation (30), following the method described in Section 3.1, but now using the principal stretch based definitions of the normal stress (33) and shear stress (36). Similar to Section 3.1, we plot the *non-normalized* loss function in the lower triangle and the *normalized* loss function in the upper triangle. Again, by design, all loss functions take a minimum of $L(\boldsymbol{\theta}; \lambda, \gamma) = 0$ for the *exact solution*, $w_i = 1$ and $w_j = 1$, indicated through the white circles. From this minimum, both versions of the loss function, non-normalized and normalized, increase with both decreasing and increasing weights w_i and w_j, and remain *convex* within the entire domain, for all 28 two-term models. Notably, in contrast to the loss function contours of the invariant based model in Figure 3, the contours of the principal stretch based model in Figure 5 display less variation for different pairs of weights w_i and w_j: Only a few pairs of weights generate loss functions of ellipsoidal shape, for example, the $\{\, w_3, w_6 \,\}$, $\{\, w_4, w_7 \,\}$ pairs in the non-normalized lower triangle, or the $\{\, w_2, w_6 \,\}$, $\{\, w_3, w_7 \,\}$, $\{\, w_4, w_8 \,\}$ pairs in the normalized upper and non-normalized lower triangles, suggesting that, in the studied stretch and shear range, only a few pairs of terms are *non-collinear* and would represent a solid base for a potential constitutive model. Most pairs of weights generate loss functions with long ridges parallel to the parameter axes, suggesting that many terms are almost *collinear* and not well suited as a functional base for a constitutive model. In contrast to the invariant based network, normalization does not seem to fix this issue, both the upper and lower triangle display this collinearity. Notably, the $\{\, w_1, w_5 \,\}$ model in the first row and fifth column and the $\{\, w_5, w_1 \,\}$ model in the fifth row and first column combine the positive and negative second powers of the principal stretches, $[\, \lambda_1^{+2} + \lambda_2^{+2} + \lambda_3^{+2} - 3 \,]$ and $[\, \lambda_1^{-2} + \lambda_2^{-2} + \lambda_3^{-2} - 3 \,]$, and represent the popular Mooney Rivlin model,[59,60] which is identical to the $\{\, w_1, w_5 \,\}$ and $\{\, w_5, w_1 \,\}$ models of the invariant based model in Figure 3. Overall, while these contours are difficult to interpret, we can compare them directly to Figure 3 and realize that, within the studied stretch and shear range λ and γ, and parameter window w_i, the invariant based network seems to represent a much broader spectrum of functions than the principal stretch based network for which the functional base seems to be generally more narrow and almost collinear. We also note that the loss function is highly *sensitive to normalization*: For both networks, the normalized

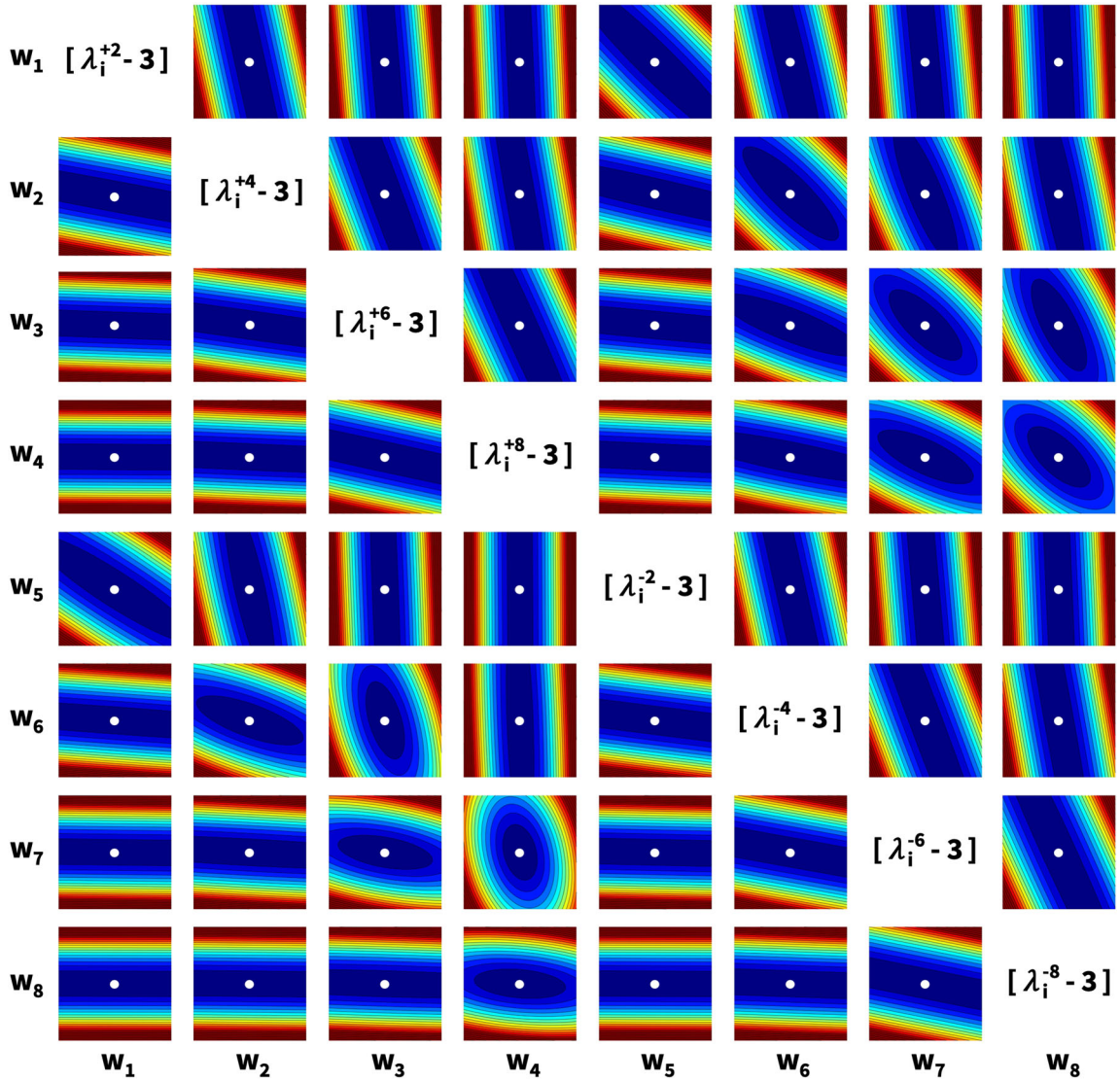

FIGURE 5 **Loss functions of principal stretch based neural network.** Contours of the loss function $L(\boldsymbol{\theta}; \lambda, \gamma)$ for all 28 possible two-term models of the principal stretch based constitutive neural network in Figure 4. The loss function is evaluated across tensile stretches $\lambda = [1.0, \ldots, 2.0]$, compressive stretches $\lambda = [1.0, \ldots, 0.5]$, and shear strains $\gamma = [0.0, \ldots, 0.5]$, with network weights in the ranges $w_i = [0, \ldots, 2]$ and $w_j = [0, \ldots, 2]$. The minimum of the loss function indicates the exact solution $w_i = 1$ and $w_j = 1$, represented through the white circle. The lower triangle illustrates the non-normalized loss function (29), the upper triangle illustrates the normalized loss function (30). All loss functions are convex, with contours varying from a few ellipsoids to many valleys with long ridges, highlighting the collinearity of many w_i and w_j pairs.

loss functions (17) and (30) tend to generate more convex shapes than the non-normalized loss functions (16) and (29), which is why we will focus on the maximum-stress normalized loss functions (17) and (30) in all following examples.

4 | L_p REGULARIZED NEURAL NETWORKS

We now integrate the concepts of L_p regularization from Section 2 and constitutive neural network modeling from Section 3 and explore the resulting regression in view of *predictability* and *interpretability*. Specifically, we supplement the loss function of the constitutive neural network with a penalty term of L_p type,

$$L(\boldsymbol{\theta}; \boldsymbol{F}) = \frac{1}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} || \boldsymbol{P}(\boldsymbol{F}_i, \boldsymbol{\theta}) - \hat{\boldsymbol{P}}_i ||^2 + \alpha || \boldsymbol{\theta} ||_p^p \rightarrow \min \quad \text{with} \quad || \boldsymbol{\theta} ||_p^p = \sum_{i=1}^{n_{\text{para}}} | w_i |^p. \tag{37}$$

The loss function minimizes the error between the model stress that we derive from the free energy of the neural network, $\boldsymbol{P}(\boldsymbol{F}_i;\boldsymbol{\theta}) = \partial\psi/\partial\boldsymbol{F}$, and the experimentally measured stress $\hat{\boldsymbol{P}}_i$ divided by the number of data points n_{data}, penalized by the L_p norm, $||\,\boldsymbol{\theta}\,||_p^p = \sum_{i=1}^{n_{\text{para}}} |\,w_i\,|^p$, of the parameter vector $\boldsymbol{\theta} = \{\ w_i\ \}$ made up of the network weights w_i, multiplied by the penalty parameter $\alpha \geq 0$. Specifically, we use tension, compression, and shear data and specify the stress error as the normalized difference between the tensile, compressive, and shear stresses predicted by the neural network, $P_{\text{ten}}(\lambda_i)$, $P_{\text{com}}(\lambda_i)$, and $P_{\text{shr}}(\gamma_i)$, and the data, $\hat{P}_{\text{ten},i}$, $\hat{P}_{\text{com},i}$, and $\hat{P}_{\text{shr},i}$, at n_{ten}, n_{com}, and n_{shr} stretch and shear levels λ and γ,

$$L(\boldsymbol{\theta};\lambda,\gamma) = \frac{1}{n_{\text{ten}}}\sum_{i=1}^{n_{\text{ten}}}\left|\left|\frac{P_{\text{ten}}(\lambda_i)-\hat{P}_{\text{ten},i}}{\hat{P}_{\text{ten}}^{\text{max}}}\right|\right|^2 + \frac{1}{n_{\text{com}}}\sum_{i=1}^{n_{\text{com}}}\left|\left|\frac{P_{\text{com}}(\lambda_i)-\hat{P}_{\text{com},i}}{\hat{P}_{\text{com}}^{\text{min}}}\right|\right|^2 + \frac{1}{n_{\text{shr}}}\sum_{i=1}^{n_{\text{shr}}}\left|\left|\frac{P_{\text{shr}}(\gamma_i)-\hat{P}_{\text{shr},i}}{\hat{P}_{\text{shr}}^{\text{max}}}\right|\right|^2 + \alpha\ ||\ \boldsymbol{\theta}\ ||_p^p \rightarrow \min. \tag{38}$$

In the following, we systematically explore the sensitivity of the loss function (38) with respect to the two hyperparameters of the L_p regularization, the power p and the penalty parameter α. For illustrative purposes, we first focus on a simplified two-term model, the Mooney Rivlin model that is shared between both neural networks, before we explore both L_p regularized complete eight-term networks.

***L_p* regularized Mooney Rivlin model.** The Mooney Rivlin model[59,60] is a two-term constitutive model that is located right at the intersection of the invariant based neural network in Figure 2 and the principal stretch based network in Figure 4. Notably, it is the *only* model, for which both networks coincide. It uses the dark red term, $[\ I_1 - 3\ ] = [\ \lambda_1^{+2} + \lambda_2^{+2} + \lambda_3^{+2} - 3\ ]$, and the green term, $[\ I_1 - 2\ ] = [\ \lambda_2^{-2} + \lambda_2^{-2} + \lambda_3^{-2} - 3\ ]$, of both neural networks, and weighs them by the network weights, $w_{1,1}\,w_{2,1} = w_1$ and $w_{1,5}\,w_{2,5} = w_5$, while all other network weights are identical to zero,

$$\psi(I_1,I_2) = w_{1,1}\ w_{2,1}\ [I_1-3] + w_{1,5}\,w_{2,5}\ [I_2-3] \ \text{ and } \ \psi(\lambda_1,\lambda_2,\lambda_3) = w_1\ [\lambda_1^{+2}+\lambda_2^{+2}+\lambda_3^{+2}-3] + w_5\ [\lambda_1^{-2}+\lambda_2^{-2}+\lambda_3^{-2}-3]. \tag{39}$$

This implies that the activation of any other weight will make the invariant and principal stretch based networks drift away from one another. The Mooney Rivlin model in Equation (39) includes the one-term dark red Neo Hooke model[61] with $[\ I_1 - 3\ ] = [\ \lambda_1^{+2} + \lambda_2^{+2} + \lambda_3^{+2} - 3\ ]$ and the one-term green Blatz Ko model[58] with $[\ I_1 - 2\ ] = [\ \lambda_1^{-2} + \lambda_2^{-2} + \lambda_3^{-2} - 3\ ]$ as special cases. For the Mooney Rivlin model, the L_p regularized loss function from Equation (38) specifies to

$$L(w_1,w_5;\lambda,\gamma) = \frac{1}{n_{\text{ten}}}\sum_{i=1}^{n_{\text{ten}}}\left|\left|\frac{P_{\text{ten}}(\lambda_i)-\hat{P}_{\text{ten},i}}{\hat{P}_{\text{ten}}^{\text{max}}}\right|\right|^2 + \frac{1}{n_{\text{com}}}\sum_{i=1}^{n_{\text{com}}}\left|\left|\frac{P_{\text{com}}(\lambda_i)-\hat{P}_{\text{com},i}}{\hat{P}_{\text{com}}^{\text{min}}}\right|\right|^2 + \frac{1}{n_{\text{shr}}}\sum_{i=1}^{n_{\text{shr}}}\left|\left|\frac{P_{\text{shr}}(\gamma_i)-\hat{P}_{\text{shr},i}}{\hat{P}_{\text{shr}}^{\text{max}}}\right|\right|^2 + \alpha_p\,[w_1^p + w_5^p] \rightarrow \min, \tag{40}$$

with the Mooney Rivlin stresses in tension, $P_{\text{ten}} = P_{11}$ for $\lambda > 1$, and compression, $P_{\text{com}} = P_{11}$ for $\lambda < 1$, from Equations (20) and (33), and in shear, $P_{\text{shr}} = P_{12}$ for all γ. from Equations (23) and (36),

$$P_{11} = 2\ [\lambda - 1/\lambda^2][w_1 + w_5/\lambda] \qquad \text{and} \qquad P_{12} = 2\ \gamma\ [w_1 + w_5]. \tag{41}$$

Notably, the uniaxial stress P_{11} and shear stress P_{12} of the Mooney Rivlin model (41) are linear functions in the network weights w_1 and w_5, which translates the neural network loss, $\sum_{i=1}^{n_{\text{ten}}}||[P_{\text{ten}}(\lambda_i) - \hat{P}_{\text{ten},i}]/P_{\text{ten}}^{\text{max}}||^2/n_{\text{ten}} + \sum_{i=1}^{n_{\text{com}}}||[P_{\text{com}}(\lambda_i) - \hat{P}_{\text{com},i}]/P_{\text{com}}^{\text{min}}||^2/n_{\text{com}} + \sum_{i=1}^{n_{\text{shr}}}||[P_{\text{shr}}(\lambda_i) - \hat{P}_{\text{shr},i}]/P_{\text{shr}}^{\text{max}}||^2/n_{\text{shr}}$, of the loss function (40) into a linear regression problem, with a single unique global minimum.

Figure 6 illustrates the contours of the L_p regularized loss function $L(w_1,w_5;\lambda,\gamma)$ for the two-term Mooney Rivlin model, with varying powers, $p = [0.25, 0.5, 0.75, 1, 1.5, 2, 4, 8]$, evaluated for the two parameters w_1 and w_5, using synthetic data from tension, compression, and shear tests. The loss function consists of the neural network loss, $\sum_{i=1}^{n_{\text{ten}}}||[P_{\text{ten}}(\lambda_i) - \hat{P}_{\text{ten},i}]/P_{\text{ten}}^{\text{max}}||^2/n_{\text{ten}} + \sum_{i=1}^{n_{\text{com}}}||[P_{\text{com}}(\lambda_i) - \hat{P}_{\text{com},i}]/P_{\text{com}}^{\text{min}}||^2/n_{\text{com}} + \sum_{i=1}^{n_{\text{shr}}}||[P_{\text{shr}}(\lambda_i) - \hat{P}_{\text{shr},i}]/P_{\text{shr}}^{\text{max}}||^2/n_{\text{shr}}$, illustrated in the first row and fifth column of Figures 3 and 5; supplemented by the L_p regularization, $\alpha_p\ [\ |\ w_1\ |^p + |\ w_5\ |^p\ ]$, illustrated in Figure 1. For all eight graphs in Figure 6, we evaluate the loss function (40) using the Mooney Rivlin stresses (41). First, we generate synthetic data, $\hat{P}_{\text{ten}}$, $\hat{P}_{\text{com}}$, $\hat{P}_{\text{shr}}$ for tensile stretches of $\lambda = [1.0, \ldots, 2.0]$, compressive stretches of $\lambda = [1.0, \ldots, 0.5]$, and shear strains of $\gamma = [0.0, \ldots, 0.5]$, in ten equidistant increments each, assuming an exact solution with $w_{1,1}\,w_{2,1} = w_1 = 1$ and $w_{1,5}\,w_{2,5} = w_5 = 1$, while fixing the remaining weights equal to zero. This results in the training data sets of eleven stretch-stress pairs for tension, compression, and shear. Second, we vary the two Mooney Rivlin network weights in the ranges $w_1 = [0, \ldots, 2]$ and $w_5 = [0, \ldots, 2]$, and evaluate the tensile, compressive, and shear model stresses, $P_{\text{ten}}(\lambda_i)$, $P_{\text{com}}(\lambda_i)$, $P_{\text{shr}}(\gamma_i)$ using Equation (41). Third, we evaluate the loss function (40) as the

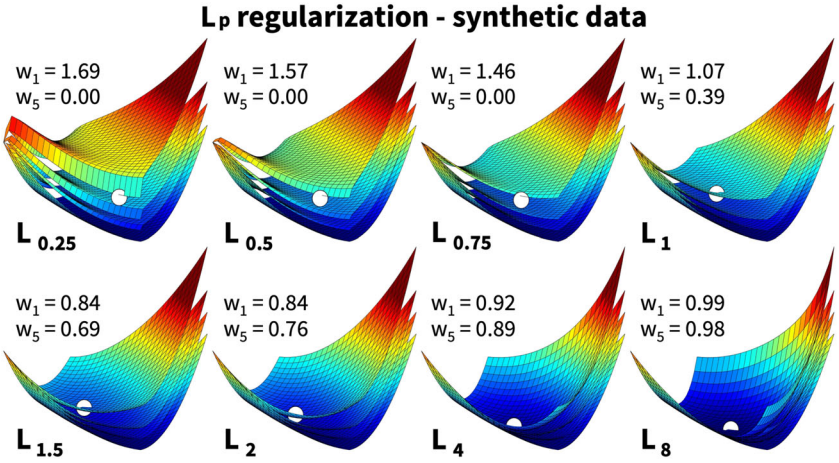

FIGURE 6 **Loss functions of L_p regularized Mooney Rivlin model for synthetic data.** Contours of the L_p regularized loss function, $L(w_1, w_5; \lambda, \gamma)$, for the two-term Mooney Rivlin model with varying powers, $p = [0.25, 0.5, 0.75, 1, 1.5, 2, 4, 8]$, evaluated for the two parameters, w_1 and w_5, for synthetic data from tension, compression, and shear tests. For $p \leq 1$, top row, with the special case of L_1 regularization or lasso in the fourth column, L_p regularization promotes sparsity by training $w_1 = 0$ exactly to zero, but the loss function is no longer strictly convex and has multiple local minima. For $p > 1$, bottom row, with the special case of L_2 regularization or ridge regression in the second column, L_p regularization promotes stability, retains both non-weights, $w_1 > 0$ and $w_5 > 0$, and maintains a convex loss function with a single global minimum.

normalized mean squared error between the model stresses $P_{\text{ten}}(\lambda_i)$, $P_{\text{com}}(\lambda_i)$, $P_{\text{shr}}(\gamma_i)$ and the synthetically generated data stresses $\hat{P}_{\text{ten}}$, $\hat{P}_{\text{com}}$, $\hat{P}_{\text{shr}}$, supplemented by the L_p regularization α_p [$\mid w_1 \mid^p + \mid w_5 \mid^p$] for the eight different powers, $p = [0.25, 0.5, 0.75, 1, 1.5, 2, 4, 8]$. As these powers p increase by two orders of magnitude, fixing the second hyperparameter α to one and the same value for all eight examples would increasingly emphasize the L_p regularization over minimizing the actual network loss, and generate increasingly biased results. Instead, for each power p, we select the penalty parameter α such that the maximum value of the loss function within the screened parameter window, in the dark red upper right corner, at $w_1 = 2$ and $w_5 = 2$, consists of equal contributions by the network term and the regularization term. This results in eight different penalty parameters, $\alpha = [\ 4.69, 3.94, 3.31, 2, 79, 1.97, 1, 39, 0.35, 0.02\ ]$. For each set of hyperparameters $\{p, \alpha\}$, we increase the penalty parameter in four increments, indicated through the four hyperplanes in each graph. Similar to the non-regularized loss functions of the invariant and principal stretch based networks in Figures 3 and 5, we highlight the minimum of the last of these four loss functions $L(w_i, w_j; \lambda, \gamma)$ through a white sphere. Importantly, in contrast to the *non-regularized* loss functions in Figures 3 and 5, the *regularized* loss function in Figure 6 no longer has a minimum of $L(\boldsymbol{\theta}; \lambda, \gamma) = 0$ at $w_1 = 1$ and $w_5 = 1$. Instead, the minimum of the loss function and its location in the $\{w_1, w_5\}$-space are now functions for the two hyperparameters p and α. For the eight powers and penalty parameters we used in this example, the minima of the loss function at the location of the white sphere become $\min(L) = [\ 5.83, 5.56, 5.24, 4.82, 3.25, 2.22, 0.57, 0.04\ ]$, and their varying locations in the $\{w_1, w_5\}$-space are indicated through the white spheres in Figure 6.

Figure 6 reveals several interesting features of the L_p regularized Mooney Rivlin model: most notably, the regularized loss function is highly sensitive to the power p and varies significantly for p *below* and *above* one, as we conclude from the different shapes in the first and second rows. For $p \leq 1$, in the top row, with the special case of L_1 regularization or lasso in the fourth column, L_p regularization promotes sparsity by training one of the weights exactly to zero, in this case $w_5 = 0$, while the other weight remains positive, $w_1 > 0$. Importantly, for $p < 1$, the loss function is no longer convex and has two local minima, one at $w_1 = 0$ and one at $w_5 = 0$. Notably, for a too small power, for example, for $p < 0.25$, we observe a drastic regularization with sharp-contoured gradients towards the parameter planes, and the model loses robustness. For $p > 1$, in the bottom row, with the special case of L_2 regularization or ridge regression in the second column, L_p regularization promotes stability and retains both non-zero weights, $w_1 > 0$ and $w_5 > 0$. The loss function remains convex with a single global minimum. Increasing the penalty parameter α amplifies these effects and moves the regularized minimum further away from the non-regularized minimum. Taken together, while a regularization across a continuous spectrum of powers p provides a lot of flexibility, the discovered weights w_1 and w_5 are highly sensitive to the

selection of the two hyperparameters p and α: while the power p acts as a switch between *sparsity* and *robustness*, the penalty parameter α induces a trade-off between *regularization* and *bias*.

Figure 7 illustrates the contours of the L_p regularized loss function $L(w_1, w_5; \lambda, \gamma)$ for the two-term Mooney Rivlin model, with varying powers, $p = [0.25, 0.5, 0.75, 1, 1.5, 2, 4, 8]$, and penalty parameters, $\alpha = [0, 0.25, 0.50, 0.75, 1, 2, 4, \infty]$, evaluated for the two parameters, w_1 and w_5, using synthetic data from tension, compression, and shear tests. For all 64 contour plots, we evaluate the normalized loss function (40) following the method of Figure 6, but now by varying both hyperparameters, p and α. Without regularization, left column, with $\alpha = 0$, all eight contour plots are identical to the non-regularized Mooney Rivlin loss function in the first row and fifth column of Figures 3 and 5. Its minimum is identical to the exact solution, $w_1 = 1$ and $w_5 = 1$, represented through the white circles. With infinite regularization, right column, with $\alpha = \infty$, all eight contour plots are a two-dimensional projection of the L_p regularization contours in Figure 1. For $p \leq 1$, in the four top rows, with increasing α, from left to right, the loss function gradually loses strict convexity, the

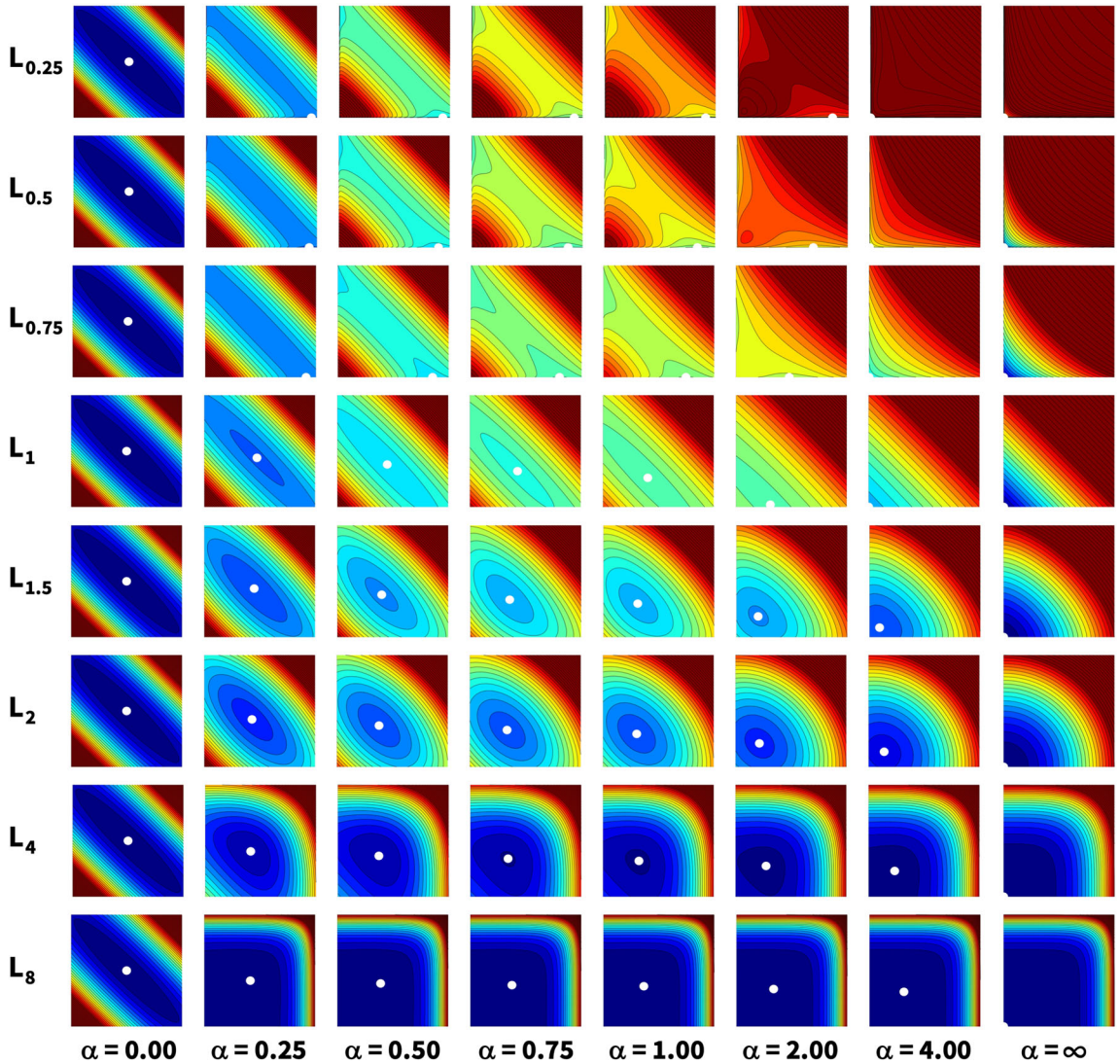

FIGURE 7 **Loss functions of L_p regularized Mooney Rivlin model for synthetic data.** Contours of the L_p regularized loss function, $L(w_1, w_5; \lambda, \gamma)$, for the two-term Mooney Rivlin model with varying powers, $p = [0.25, 0.5, 0.75, 1, 1.5, 2, 4, 8]$, and penalty parameters $\alpha = [0, 0.25, 0.50, 0.75, 1, 2, 4, \infty]$, evaluated for the two parameters, w_1 and w_5, for synthetic data from tension, compression, and shear tests. Without regularization, left column, with $\alpha = 0$, the minimum of the loss function is identical to the exact solution $w_1 = 1$ and $w_5 = 1$, represented through the white circle. With infinite regularization, right column, with $\alpha = \infty$, the loss function is identical to the L_p regularization term and the contours are identical to Figure 1. For $p < 1$, with increasing α, the loss function gradually loses convexity, the minimum first moves towards $w_1 \geq 1$ and $w_5 = 0$, and then towards $w_1 \to 0$ and $w_5 = 0$. For $p > 1$, with increasing α, the loss function always remains convex, both weights always remain active, $w_1 \geq 0$ and $w_5 \geq 0$, as the minimum moves gradually towards $w_1 \to 0$ and $w_5 \to 0$.

minimum first moves towards $w_1 \geq 1$ and $w_5 = 0$, and then towards $w_1 \to 0$ and $w_5 = 0$. For $p > 1$, in the four bottom rows, with increasing α, from left to right, the loss function always remains convex, both weights always remain active, $w_1 \geq 0$ and $w_5 \geq 0$, and move closer together as the minimum gradually moves towards zero, $w_1 \to 0$ and $w_5 \to 0$.

Figure 7 confirms our observations from Figure 6 and provides additional insights into the L_p regularized Mooney Rivlin model: the regularized loss function is highly sensitive to both hyperparameters, p and α. Decreasing the power to or below one, $p \leq 1$, *increases interpretability* by promoting sparsity as a subset of weights become exactly zero; smaller powers p and larger penalty parameters α promote sparsity more drastically and generate increasingly less convex loss functions. Increasing the power above one, $p > 1$, *increases predictability* by promoting robustness as the loss function becomes increasingly convex; larger powers p and larger penalty parameters α promote robustness more drastically and generate increasingly more convex loss functions. These observations confirm the general notion that L_p regularization is an intricate balance between predictability and interpretability and between regularization and bias that requires a careful selection of the appropriate values for the hyperparameters p and α.

Figure 8 illustrates the contours of the L_p regularized loss function $L(w_1, w_5; \lambda, \gamma)$ for the two-term Mooney Rivlin model, with varying powers p, evaluated for the two parameters w_1 and w_5, but now using real data from tension, compression, and shear tests of human brain.[62] For all eight graphs, we evaluate the loss function (40) as the normalized mean squared error between the model $P_{\text{ten}}(\lambda_i)$, $P_{\text{com}}(\lambda_i)$, $P_{\text{shr}}(\gamma_i)$, and the data $\hat{P}_{\text{ten}}$, $\hat{P}_{\text{com}}$, $\hat{P}_{\text{shr}}$, for tensile stretches of $\lambda = [1.0, \ldots, 1.1]$, compressive stretches of $\lambda = [0.9, \ldots, 1.0]$, and shear strains of $\gamma = [0.0, \ldots, 0.2]$, in 16 equidistant increments each,[29] for varying Mooney Rivlin network weights in the ranges $w_1 = [0, \ldots, 1]$ and $w_5 = [0, \ldots, 1]$, and apply L_p regularization, $\alpha_p \, [\, | \, w_1 \, |^p + | \, w_5 \, |^p \,]$, for eight different powers, $p = [0.25, 0.5, 0.75, 1, 1.5, 2, 4, 8]$. We select a penalty parameter $\alpha^{\max} = 0.6585$, such that the maximum value of the loss function within the screened parameter window, in the dark red upper right corner, at $w_1 = 1$ and $w_5 = 1$, consists of equal contributions by the network term and the regularization term. For each set of hyperparameters $\{p, \alpha\}$, we increase the penalty parameter in four increments, $\alpha = [0.00, 0.25, 0.50, 1.00]\alpha^{\max}$, indicated through the four hyperplanes in each graph, and highlight the minimum of the last of these four loss functions $L(w_i, w_j; \lambda, \gamma)$ through a white sphere. Similar to Figure 6 based on synthetic data, the minimum of the loss function and its location in the $\{w_1, w_5\}$-space are functions for the two hyperparameters p and α. For the plain non-regularized loss function, the minimum of the loss function is 0.0713 and its weights are $w_1 = 0.00$ and $w_5 = 0.84$. For the eight powers, the minima of the loss function at the location of the white sphere are $\min(L) = [\, 0.67, 0.63, 0.56, 0.50, 0.34, 0.24, 0.11, 0.08 \,]$, and their varying locations in the $\{w_1, w_5\}$-space are indicated through the white spheres in Figure 8.

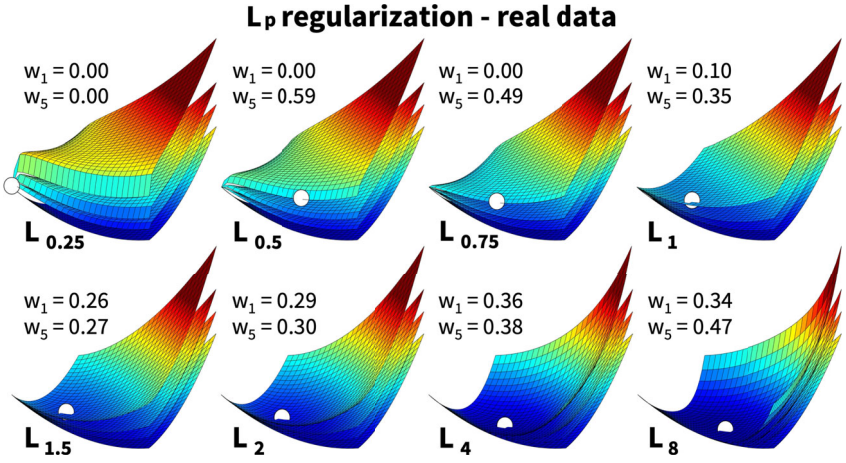

FIGURE 8 **Loss functions of L_p regularized Mooney Rivlin model for real data.** Contours of the L_p regularized loss function, $L(w_1, w_5; \lambda, \gamma)$, for the two-term Mooney Rivlin model with varying powers, $p = [0.25, 0.5, 0.75, 1, 1.5, 2, 4, 8]$, evaluated for the two parameters, w_1 and w_5, for real data from tension, compression, and shear tests of human brain. For $p \leq 1$, top row, with the special case of L_1 regularization or lasso in the fourth column, L_p regularization promotes sparsity by training $w_1 = 0$ exactly to zero, but the loss function is no longer strictly convex and has multiple local minima. For $p > 1$, bottom row, with the special case of L_2 regularization or ridge regression in the second column, L_p regularization promotes stability, retains both non-weights, $w_1 > 0$ and $w_5 > 0$, and maintains a convex loss function with a single global minimum.

Figure 8 reveals several interesting differences between the loss function for *synthetic data* in Figure 6 and for *real data*, in this case from human brain experiments, in Figure 8. Most importantly, for the synthetic data, we assumed an *exact* minimum at $w_1 = 1$ and $w_5 = 1$, where the loss function is exactly zero for the non-L_p-regularized model, and takes the value of the regularization term $\alpha \, || \, \theta \, ||_p^p$ otherwise. For the real data, we no longer know a priori where the exact minimum is and it is no longer exactly zero, since the Mooney Rivlin model is not exact for the real data. From screening the parameter plane, find the minimum loss at $w_1 = 0.00$ and $w_5 = 0.84$. Strikingly, this suggests that the one-parameter Blatz Ko model[58] with $\psi = w_5 \, [\, I_2 - 3 \,]$ and $P = w_5 \, \partial I_2 / \partial \boldsymbol{P} - p \, \boldsymbol{P}^{-t}$ is better suited to describe the experimental data than the two-parameter Mooney Rivlin[59,60] model. However, we can clearly see the negative effect of *over-regularization* with too large penalty parameters α: for $p < 1$, in the top row, the minimum of the loss function remains on the $w_1 = 0.00$ axis, but the Blatz Ko parameter is drastically reduced from its non-regularized value of $w_5 = 0.84$ to $w_5 = 0.59$, $w_5 = 0.49$, and even $w_5 = 0$. For $p \geq 1$, in the bottom row, the minimum of the loss function even moves away from the $w_1 = 0.00$ axis, and both parameters become activated at a similar magnitude between $w_1 = [0.26, \, \dots \, , 0.36]$ and $w_5 = [0.27, \, \dots \, , 0.47]$. Taken together, the discovered weights w_1 and w_5 are highly *sensitive to over-regularization* for extreme ranges of the hyperparameters p and α: extreme penalty parameters induce increased *bias* as the loss function increasingly focuses on minimizing the penalty term rather than the regularization problem itself.

4.1 | L_p regularized invariant based neural network

Similar to the previous example, we explore the effects of L_p regularization with respect to the two hyperparameters p and α, but now for the full eight-term invariant based network,[29] instead of the two-term Mooney Rivlin model,[59,60] and for training on real instead of synthetic data. We use tension, compression, and shear data from human brain tests,[62] over a tensile range of $\lambda = [1.0, \, \dots \, , 1.1]$, a compressive range of $\lambda = [0.9, \, \dots \, , 1.0]$, and a shear range of $\gamma = [0.0, \, \dots \, , 0.2]$, sampled in 16 equidistant increments each, averaged over anywhere between $n = 15$ and $n = 35$ specimen.[29] We train the invariant based neural network from Figure 2 in Section 3.1 and minimize the loss function from Equation (38) with the stress definitions (20) and (23) with three different powers, $p = [0.5, 1.0, 2.0]$, and four different penalty parameters, $\alpha = [0.000, 0.001, 0.010, 0.100]$. We use the Adam optimizer, a robust adaptive algorithm for stochastic gradient-based first-order optimization.[68]

Figure 9 summarizes our four discovered models in terms of the nominal stress as a function of stretch or shear strain, with the penalty parameter α increasing from left to right. The circles represent the experimental data.[62] The color-coded regions represent the stress contributions of the eight model terms according to Figure 2. The coefficients of determination R^2 quantify the goodness of fit. Overall, the L_p regularized invariant based network trains solidly and provides a good fit of the data. Without regularization, in the left column, the network discovers four non-zero terms, all in terms of the second invariant, $[\, I_2 - 3 \,]$, indicated through cold green-to-blue colors,

$$\psi = w_5 \, [\, I_2 - 3 \,] + w_{2,6} \, [\, \exp(\, w_{1,6} \, [\, I_2 - 3 \,]) - 1 \,] + w_7 \, [\, I_2 - 3 \,]^2 + w_{2,8} \, [\, \exp(\, w_{1,8} \, [\, I_2 - 3 \,]^2) - 1 \,],$$

with stiffness-like parameters $w_5 = 0.129 \, \text{kPa}$, $w_{2,6} = 0.358 \, \text{kPa}$, $w_7 = 3.840 \, \text{kPa}$, and $w_{2,8} = 1.406 \, \text{kPa}$ and exponential weights, $w_{1,6} = 1.152$ and $w_{1,8} = 2.891$, and its stress takes the following form, $\boldsymbol{P} = [w_5 + w_{2,6} w_{1,6} \exp(\, w_{1,6} [\, I_2 - 3 \,]) + 2 \, [\, I_2 - 3 \,][w_7 + w_{2,8} w_{1,8} \exp(\, w_{1,8} [\, I_2 - 3 \,]^2)]] \partial I_2 / \partial \boldsymbol{F} - p \, \boldsymbol{F}^{-t}$. As the penalty parameter increases, from left to right, the number of non-zero terms decreases. With a penalty parameter $\alpha = 0.010$, in the third column, the network discovers three non-zero terms, all in terms of the second invariant, $[\, I_2 - 3 \,]$, indicated through cold green-to-light-blue colors,

$$\psi = w_5 \, [\, I_2 - 3 \,] + w_{2,6} \, [\, \exp(\, w_{1,6} \, [\, I_2 - 3 \,]) - 1 \,] + w_7 \, [\, I_2 - 3 \,]^2,$$

with stiffness-like parameters $w_5 = 0.231 \, \text{kPa}$, $w_{2,6} = 1.443 \, \text{kPa}$, and $w_7 = 7.364 \, \text{kPa}$, and the exponential weight, $w_{1,6} = 5.102$, and its stress takes the following form, $\boldsymbol{P} = [w_5 + w_{2,6} w_{1,6} \exp(\, w_{1,6} [\, I_2 - 3 \,]) + 2 \, [\, I_2 - 3 \,] w_7] \partial I_2 / \partial \boldsymbol{F} - p \, \boldsymbol{F}^{-t}$. For the largest penalty parameter, in the right column, the network discovers a single non-zero term, the turquoise linear exponential term of the second invariant,

$$\psi = w_{2,6} \, [\, \exp(\, w_{1,6} \, [\, I_2 - 3 \,]) - 1 \,],$$

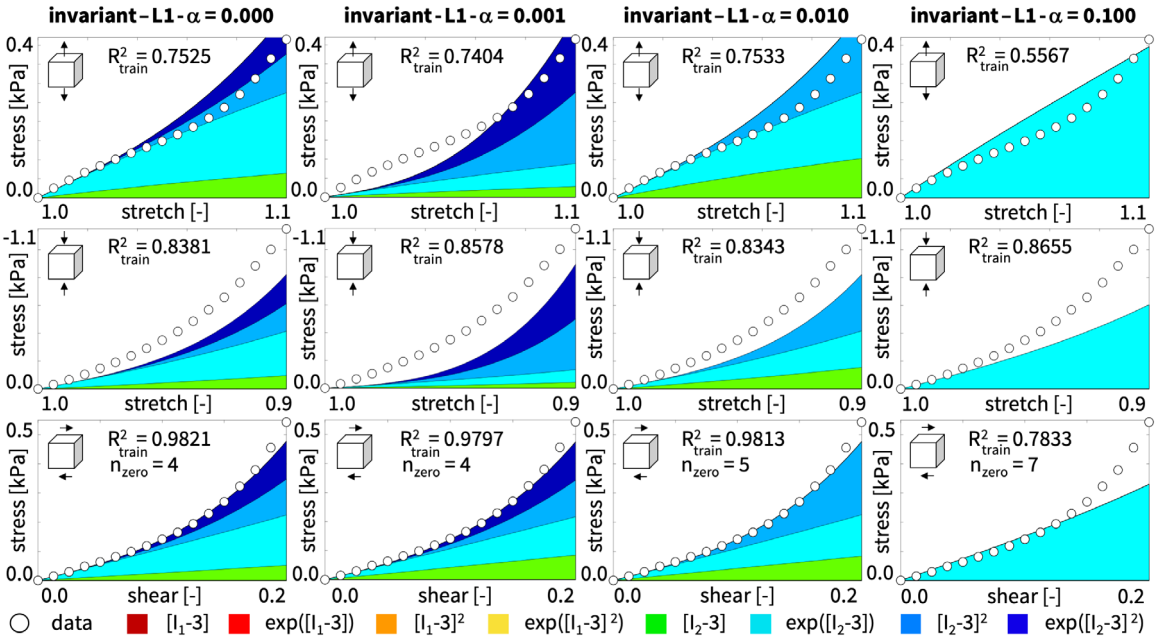

FIGURE 9 **Discovered models of L_1 regularized invariant based network.** Nominal stress as a function of stretch or shear strain for the invariant based neural network with L_1 regularization for varying penalty parameters $\alpha = [\ 0.000, 0.001, 0.010, 0.100\ ]$, trained with human gray matter tension, compression, and shear data. Circles represent the experimental data. Color-coded regions represent the discovered model terms. Coefficients of determination R^2 indicate the goodness of fit.

with the stiffness-like parameter $w_{2,6} = 0.2462\,\text{kPa}$ and the exponential weight $w_{1,6} = 2.9937$, and its stress takes the following form, $\boldsymbol{P} = w_{2,6}\ w_{1,6}\ \exp(\ w_{1,6}\ [\ I_2 - 3\ ])\ \partial I_2/\partial \boldsymbol{F} - p\ \boldsymbol{F}^{-t}$. While Figure 9 provides great visual insights into the performance of L_1 regularization with varying penalty parameters, it only represents a *snapshot of model discovery* in the eight-dimensional parameter space of the network. Subset selection and model discovery are not only sensitive to the initialization of the parameter vector $\boldsymbol{\theta} = \{\ w_i\ \}$, but also to the stochastic nature of the Adam optimizer. This implies that different runs may produce different results. This raises the question how *reproducible* and *robust* the results in Figure 9 are for varying initial conditions and training runs.

Figure 10 summarizes the discovered weights for the invariant based network with L_p regularization for varying powers $p = [\ 0.5, 1.0, 2.0]$ and penalty parameters $\alpha = [\ 0.000, 0.001, 0.010, 0.100\ ]$. For all twelve combinations of the two hyperparameters, we perform a total of $n = 100$ training runs each, with varying initial conditions for the network weights $w_i = \{\ w_1, \dots, w_8\ \}$, such that each of the four models in Figure 9 is the result of one of the L_1 regularized training runs in the middle row. The colored boxes in Figure 10 indicate the relevance of the eight model terms, with their means and standard deviations. Interestingly, the $L_{0.5}$ and $L_{0.1}$ regularizations in the first and second rows perform qualitatively similarly: they both start with four dominant terms, all in terms of the second invariant. Except for a small number of outliers, they both converge to two dominant one-term models, the green $[\ I_2 - 3\ ]$ and the turquoise $\exp([\ I_2 - 3\ ])$ models, while all other weights train to zero. The fact that both networks *alternate* between these two terms is a result of the *non-convex* nature of the underlying nonlinear regression problem associated with the invariant based network and indicates the existence of *multiple local minima*. Instead, the L_2 regularization in the bottom row converges to a model that consistently trains the dark red $[\ I_1 - 3\ ]$ and red $\exp([\ I_1 - 3\ ])$ terms to zero, and maintains six non-zero terms, of which the yellow $\exp([\ I_1 - 3\ ]^2)$, turquoise $\exp([\ I_2 - 3\ ])$, and dark blue $\exp([\ I_2 - 3\ ]^2)$ terms are dominant.

Figure 11 summarizes the convergence of the L_p regularized invariant based network in terms of the goodness of fit and number of terms, for varying powers $p = [\ 0.5, 1.0, 2.0\ ]$ and penalty parameters $\alpha = [\ 0.000, 0.001, 0.005, 0.010, 0.015, 0.020, 0.040, 0.060, 0.080, 0.100\ ]$. Red dots indicate the coefficient of determination R^2, blue dots indicate the number of terms, with means and standard deviations from $n = 10$ realizations. A known shortcoming of the L_p regularization is that it introduces bias and moves the solution away from the minimum of the network loss towards the minimum of the regularization loss. This is particularly critical for our network in which all weights have a different meaning and potentially also a different magnitude. To quantify the effects of this potential limitation, Figure 11 compares the non-normalized regularization, $||\ \boldsymbol{\theta}\ ||_p^p = \sum_{i=1}^{n_{\text{para}}} |\ w_i\ |^p$, in terms of the weights w_i that we have used

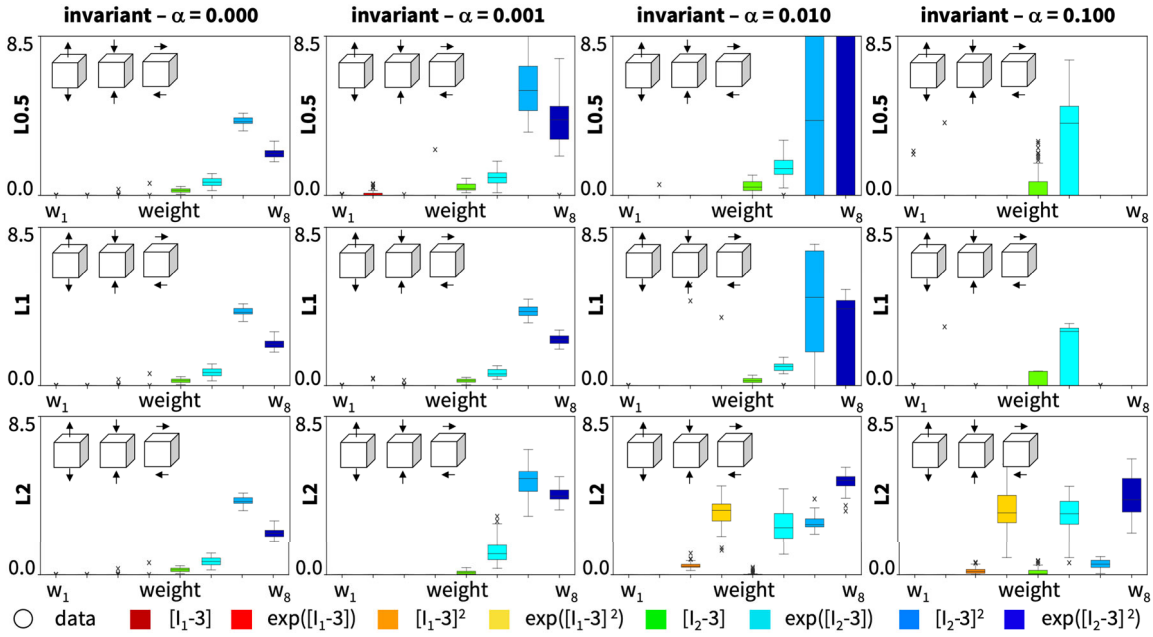

FIGURE 10 **Discovered models of L_p regularized invariant based network.** Distribution of discovered weights for the invariant based neural network with L_p regularization for varying powers $p = [\,0.5, 1.0, 2.0]$ and penalty parameters $\alpha = [\,0.000, 0.001, 0.010, 0.100\,]$. Colored boxes indicate the relevance of the eight model terms, with means and standard deviations from $n = 100$ realizations with varying initializations of the network weights.

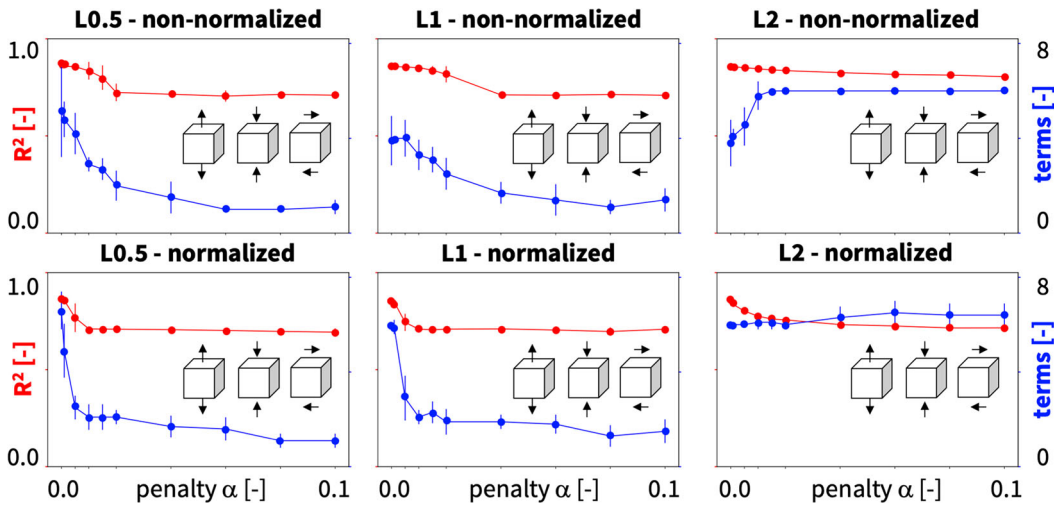

FIGURE 11 **Convergence of L_p regularized invariant based network.** Goodness of fit and number of terms for the invariant based neural network with L_p regularization for varying powers $p = [\,0.5, 1.0, 2.0\,]$ and penalty parameters $\alpha = [\,0.000, 0.001, 0.005, 0.010, 0.015, 0.020, 0.040, 0.060, 0.080, 0.100\,]$. Top row uses a non-normalized regularization in terms of the weights w_i, bottom row uses a normalized regularization in terms of the L_0-normalized weights $w_i \,/\, w_{i,L_0}$. Red dots indicate the coefficient of determination R^2, blue dots indicate the number of terms, with means and standard deviations from $n = 10$ realizations.

throughout this study against a normalized regularization, $|| \boldsymbol{\theta} ||_p^p = \sum_{i=1}^{n_{\text{para}}} | w_i \,/\, w_{i,L_0} |^p$, in terms of the L_0-normalized weights $w_i \,/\, w_{i,L_0}$. Here, w_{i,L_0} are the weights of the one-term models from the diagonal in Table 1.

Figure 11 confirms that regularization is a trade-off between *error* and *complexity*, or similarly, between the *goodness of fit* and the *number of terms*. While the $L_{0.5}$ and L_1 regularizations behave qualitatively similarly and promote sparsity by reducing the number of non-zero terms to one, the L_2 promotes robustness by maintaining a large subset of six non-zero terms. The L_1 regularization is less aggressive than the $L_{0.5}$ regularization and requires larger penalty parameters α to achieve a similar sparseness, which could induce a larger bias, away from minimum of the network loss towards the minimum of the regularization loss. Normalizing the L_p penalty term by using the L_0-normalized weights $w_i \,/\, w_{i,L_0}$ instead of the non-normalized weights w_i accelerates the positive effects of regularization, especially in the small-penalty-parameter regime, and could provide a viable solution to reduce regularization-induced bias. Ultimately, in the large-penalty-parameter regime, the non-normalized and normalized regularizations converge towards a similar goodness of fit and number of terms.

Taken together, our results confirm the general notion that L_p regularization *increases interpretability* for powers equal to or below one, $p \leq 1$, by promoting sparsity as a subset of weights train exactly to zero; and *increases predictability* for powers larger than one, $p > 1$, by promoting robustness as a unique subset of weights emerges as dominant. Larger penalty parameters α amplify these trends at the price of an increased bias, which we can reduce, at least in part, by normalizing the network weights in the the penalty term.

L_0 regularized invariant based neural network. For comparison, we explore the effects of L_0 regularization using the same tension, compression, and shear data from human brain tests as in the previous example.[62] We train the invariant based neural network from Figure 2 in Section 3.1 and minimize the loss function from Equation (38) with the stress definitions (20) and (23), but now use a penalty term, $\alpha \, || \, \boldsymbol{\theta} \, ||_0$, with the L_0 norm $|| \boldsymbol{\theta} ||_0 = \sum_{i=1}^{n_{\text{para}}} I(w_i \neq 0)$, to penalize the total number of non-zero terms in the model. In essence, L_0 regularization turns network training into a *discrete combinatorial problem* with $2^8 - 1 = 255$ possible models, 8 with a single term, 28 with two, 56 with three, 70 with four, 56 with five, 28 with six, 8 with seven, and 1 with all eight terms. For illustrative purposes, we focus on the eight *one-term* and 28 *two-term models* that follow by explicitly setting the other seven and six terms of the network to zero.

Table 1 summarizes the weights and remaining losses of the one- and two-term models of the L_0 regularized invariant based neural network. The diagonal summarizes the discovered one-term models, the off-diagonal the two-term models. The L_0 regularization penalizes the one-term models by α and the two-term models by 2α. The boldface cells highlight the four *best-in-class models* of each category. Figures 12 and 13 illustrate the stress-stretch and stress-shear plots of these four one- and two-term models. Interestingly, all best-in-class models are models in terms of the second invariant I_2 indicated through the cold green-to-blue colors. None of the eight best models includes the first invariant I_1 indicated through the warm red-to-yellow colors. This finding contradicts the common practice of using primarily the first invariant, for example, in popular and widely used neo Hooke model. Strikingly, the classical Hooke model[61] represented through the dark red term in both networks with $\psi = w_1 \, [\, I_1 - 3 \,]$, a stiffness-like parameter $w_1 = 0.7964\,\text{kPa}$, a shear modulus $\mu = 2 \, w_1 = 1.5928\,\text{kPa}$, and a remaining loss of $0.0918 + \alpha$ has the largest remaining loss and performs *the worst* of all one-term models. Similarly, the Demiray model[69] represented through the red term with $\psi = w_{2,2} \, [\, \exp(\, w_{1,2} \, [\, I_1 - 3 \,] \,) - 1 \,]$, a stiffness-like parameter $w_{2,2} = 0.7265\,\text{kPa}$, an exponent $w_{1,2} = 1.0763$, and a remaining loss of $0.0894 + \alpha$, and the Holzapfel type model[70] represented through the yellow term with $\psi = w_{2,4} \, [\, \exp(\, w_{1,4} \, [\, I_1 - 3 \,]^2 \,) - 1 \,]$, a stiffness-like parameter $w_{2,4} = 4.2436\,\text{kPa}$, an exponent $w_{1,4} = 4.3048$, and a remaining loss of $0.0863 + \alpha$, also perform worse than all one-term second-invariant models. Yet, these results agree well with our previous observations that the second invariant is better suited to represent the behavior of brain tissue than the first invariant.[29] The best-in-class one-term model with the lowest remaining loss is the model with the light blue quadratic term of the second invariant,

$$\psi = w_7 \, [\, I_2 - 3 \,]^2,$$

with the stiffness-like parameter $w_7 = 19.5994\,\text{kPa}$ for which the stress takes the following form, $\boldsymbol{P} = 2 \, [\, I_2 - 3 \,] \, w_7 \, \partial I_2 / \partial \boldsymbol{F} - p \, \boldsymbol{F}^{-\text{t}}$. The best-in-class two-term model is the model with the turquoise linear exponential and the dark blue quadratic exponential terms of the second invariant,

$$\psi = w_{2,6} \, [\, \exp(\, w_{1,6} \, [\, I_2 - 3 \,]) - 1 \,] + w_{2,8} \, [\, \exp (\, w_{1,8} \, [\, I_2 - 3 \,])^2 - 1 \,],$$

TABLE 1 L_0 regularized invariant based neural network.

	$\mathbf{w_1}$	$\mathbf{w_{1,2}}$, $\mathbf{w_{2,2}}$	$\mathbf{w_3}$	$\mathbf{w_{1,4}}$, $\mathbf{w_{2,4}}$	$\mathbf{w_5}$	$\mathbf{w_{1,6}}$, $\mathbf{w_{2,6}}$	$\mathbf{w_7}$	$\mathbf{w_{1,8}}$, $\mathbf{w_{2,8}}$
$\mathbf{w_1}$	0.796	0.237 0.918, 0.600	0.400 10.048	0.403 3.666, 2.718	0.000 0.840	0.000 0.957, 0.865	0.330 12.545	0.330 3.810, 3.286
Loss	$0.092 + \alpha$	$0.090 + 2\alpha$	$0.060 + 2\alpha$	$0.060 + 2\alpha$	$0.071 + 2\alpha$	$0.069 + 2\alpha$	$0.040 + 2\alpha$	$0.040 + 2\alpha$
$\mathbf{w_{1,2}}$,$\mathbf{w_{2,2}}$	0.918, 0.600 0.237	1.076, 0.727	0.980, 0.410 9.811	1.219, 0.329 4.167, 2.342	0.369, 0.000 0.841	0.558, 0.000 3.186, 0.250	1.095, 0.294 12.547	0.822, 0.407 4.089, 2.993
Loss	$0.090 + 2\alpha$	$0.089 + \alpha$	$0.060 + 2\alpha$	$0.060 + 2\alpha$	$0.071 + 2\alpha$	$0.063 + 2\alpha$	$0.040 + 2\alpha$	$0.040 + 2\alpha$
$\mathbf{w_3}$	10.048 0.400	9.811 0.980, 0.410	18.348	9.388 3.011, 2.977	8.151 0.507	8.173 0.876, 0.569	8.216 10.916	8.178 3.193, 3.422
Loss	$0.060 + 2\alpha$	$0.060 + 2\alpha$	$0.086 + \alpha$	$0.086 + 2\alpha$	$0.049 + 2\alpha$	$0.049 + 2\alpha$	$0.069 + 2\alpha$	0.070
$\mathbf{w_{1,4}}$, $\mathbf{w_{2,4}}$	3.666, 2.718 0.403	4.167, 2.342 1.219, 0.329	3.011, 2.977 9.388	4.305, 4.244	3.134, 2.672 0.497	3.014, 2.669 1.266, 0.394	3.210, 2.559 10.896	2.973, 2.737 3.149, 3.477
Loss	$0.060 + 2\alpha$	$0.060 + 2\alpha$	$0.086 + 2\alpha$	$0.086 + \alpha$	$0.049 + 2\alpha$	$0.048 + 2\alpha$	$0.070 + 2\alpha$	$0.070 + 2\alpha$
$\mathbf{w_5}$	0.840 0.000	0.841 0.369, 0.000	0.507 8.151	0.497 3.134, 2.672	**0.840**	0.234 0.747, 0.803	**0.406** **11.178**	**0.411** **3.644, 3.030**
Loss	$0.071 + 2\alpha$	$0.071 + 2\alpha$	$0.049 + 2\alpha$	$0.049 + 2\alpha$	$\mathbf{0.071 + \alpha}$	$0.070 + 2\alpha$	$\mathbf{0.033 + 2\alpha}$	$\mathbf{0.033 + 2\alpha}$
$\mathbf{w_{1,6}}$,$\mathbf{w_{2,6}}$	0.957, 0.865 0.000	3.186, 0.250 0.558, 0.000	0.876, 0.569 8.173	1.266, 0.394 3.014, 2.669	0.747, 0.803 0.234	**0.898, 0.923**	**1.022, 0.396** **11.014**	**0.839, 0.484** **3.427, 3.208**
Loss	$0.069 + 2\alpha$	$0.063 + 2\alpha$	$0.049 + 2\alpha$	$0.048 + 2\alpha$	$0.070 + 2\alpha$	$\mathbf{0.069 + \alpha}$	$\mathbf{0.033 + 2\alpha}$	$\mathbf{0.033 + 2\alpha}$
$\mathbf{w_7}$	12.545 0.330	12.547 1.095, 0.294	10.916 8.216	10.896 3.210, 2.559	**11.178** **0.406**	**11.014** **1.022, 0.396**	**19.599**	9.520 3.565, 2.834
Loss	$0.040 + 2\alpha$	$0.040 + 2\alpha$	$0.069 + 2\alpha$	$0.070 + 2\alpha$	$\mathbf{0.033 + 2\alpha}$	$\mathbf{0.033 + 2\alpha}$	$\mathbf{0.059 + \alpha}$	$0.059 + 2\alpha$
$\mathbf{w_{1,8}}$, $\mathbf{w_{2,8}}$	3.810, 3.286 0.330	4.089, 2.993 0.822, 0.407	3.193, 3.422 8.178	3.149, 3.477 2.973, 2.737	**3.644, 3.030** **0.411**	**3.427, 3.208** **0.839, 0.484**	3.565, 2.834 9.520	**4.560, 4.280**
Loss	$0.040 + 2\alpha$	$0.040 + 2\alpha$	$0.070 + 2\alpha$	$0.070 + 2\alpha$	$\mathbf{0.033 + 2\alpha}$	$\mathbf{0.033 + 2\alpha}$	$0.059 + 2\alpha$	$\mathbf{0.060 + \alpha}$

Note: Weights and remaining losses of the one- and two-term models of the L_0 regularized invariant based neural network. The diagonal summarizes the discovered one-term models penalized by α, the off-diagonal the two-term models penalized by 2α. Best-in-class models are highlighted in bold.

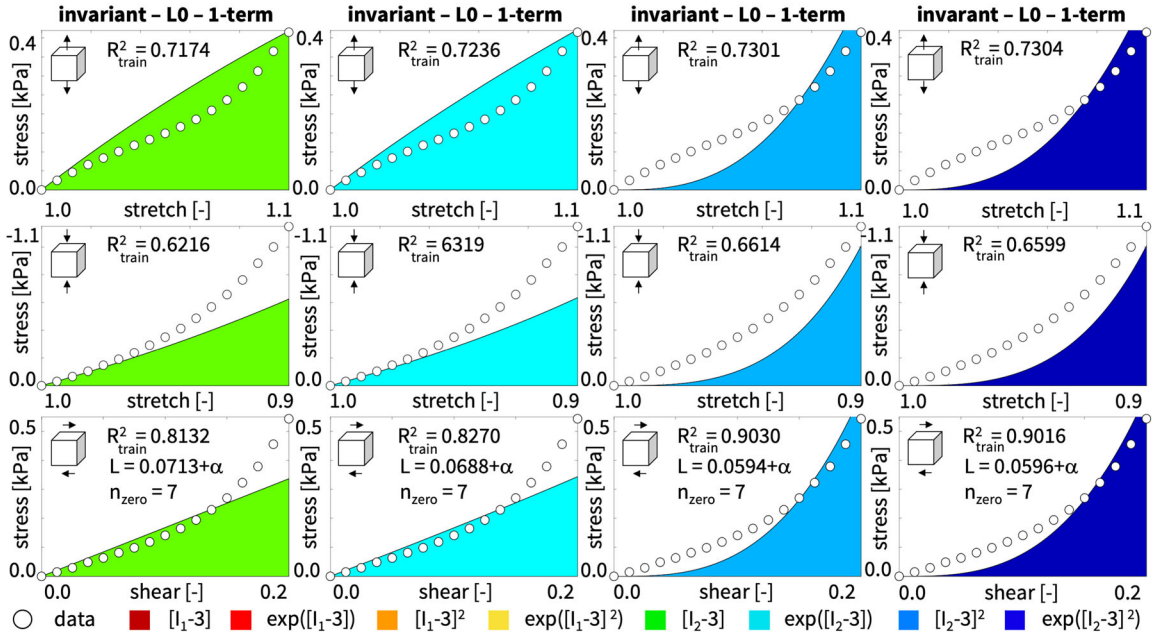

FIGURE 12 **Discovered best-in-class one-term models of L_0 regularized invariant based network.** Nominal stress as a function of stretch or shear strain for the invariant based constitutive neural network with L_0 regularization, trained with human gray matter tension, compression, and shear data. Circles represent the experimental data. Color-coded regions represent the discovered model terms. Coefficients of determination R^2 indicate the goodness of fit for each individual test; remaining loss L indicates the quality of the overall fit.

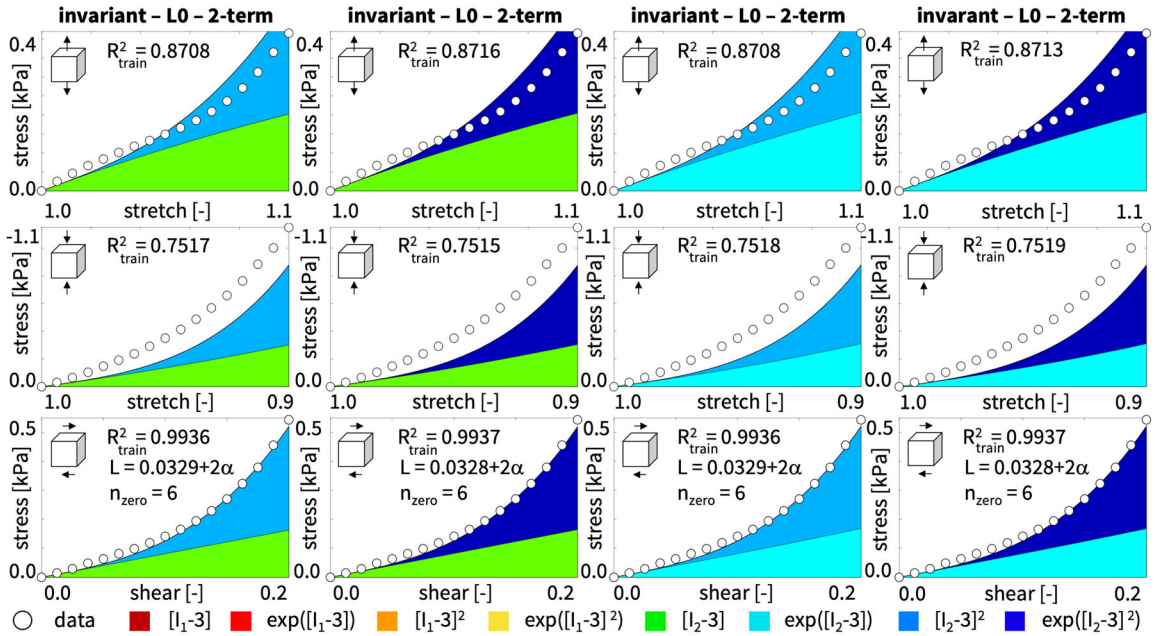

FIGURE 13 **Discovered best-in-class two-term models of L_0 regularized invariant based network.** Nominal stress as a function of stretch or shear strain for the invariant based constitutive neural network with L_0 regularization, trained with human gray matter tension, compression, and shear data. Circles represent the experimental data. Color-coded regions represent the discovered model terms. Coefficients of determination R^2 indicate the goodness of fit for each individual test; remaining loss L indicates the quality of the overall fit.

with the stiffness-like parameters $w_{2,6} = 0.4835\,\text{kPa}$ and $w_{2,8} = 3.2080\,\text{kPa}$, and the exponential weights $w_{1,6} = 0.8393$ and $w_{1,8} = 3.4273$, for which the stress takes the following form, $\boldsymbol{P} = [\ w_{2,6}\ w_{1,6}\ \exp(\ w_{1,6}\ [\ I_2 - 3\ ]) + 2\ [\ I_2 - 3\ ]\ w_{2,8}\ w_{2,8}\ \exp(\ w_{1,8}\ [\ I_2 - 3\ ])^2]\ \partial I_2/\partial \boldsymbol{F} - p\ \boldsymbol{F}^{-\text{t}}$. For this simple example, the remaining loss of the best one-term model is $0.0594 + \alpha$ and the remaining loss of the best two-term model is $0.0328 + 2\alpha$. This implies that, for penalty parameters $\alpha < 0.0266$, the L_0 regularization would favor the two-term model, while for penalty parameters $\alpha \geq 0.0266$, the L_0 regularization would favor the one-term model. These simple considerations highlight the importance of the penalty parameter α, which explicitly acts as a *discrete switch* between the number of terms we want to include in our model.

Taken together, this example illustrates the discrete nature of the L_0 regularization as a *discrete combinatorial problem* that becomes increasingly expensive as the number of model terms increases. Our results emphasize the sensitivity of the L_0 regularization with respect to the penalty parameter α and highlight the trade-off between *bias* and *variance*: increasing the penalty parameter increases bias, reduces variance, and decreases model complexity as the total number of non-zero terms decreases towards one.

4.2 | Lp regularized principal stretch based neural network

Similar to the previous example, we explore the effects of L_p regularization with respect to the two hyperparameters p and α, but now for the full principal stretch based network with all eight terms,[29] for training on tension, compression, and shear data from human brain tests.[62] We train the principal based neural network from Figure 4 in Section 3.2 and minimize the loss function from Equation (38) with the stress definitions (33) and (36) with three different powers, $p = [0.5, 1.0, 2.0]$, and four different penalty parameters, $\alpha = [0.000, 0.001, 0.010, 0.100]$ using the Adam optimizer.[68]

Figure 14 summarizes our four discovered models in terms of the nominal stress as a function of stretch or shear strain, with the penalty parameter α increasing from left to right. The circles represent the experimental data.[62] The color-coded regions represent the stress contributions of the eight model terms according to Figure 4. The coefficients of determination R^2 quantify the goodness of fit. Similar to the invariant based network in Section 4.1, the L_p regularized principal stretch based network trains solidly and provides a good fit of the data. Without regularization, in the left column, the network discovers all eight non-zero terms. As the penalty parameter increases, from left to right, the number of non-zero terms decreases. For the largest penalty parameter, in the right column, the network discovers a single dominant term, the dark blue $[\ \lambda_i^{-8} - 3\ ]$ term,

$$\psi = w_8 \sum_{i=3}^{3} [\ \lambda_i^{-8} - 1\ ]$$

with a stiffness-like parameter $w_8 = 0.0534\,\text{kPa}$ and a stress $\boldsymbol{P} = -8\ w_8\ \sum_{i=3}^{3} [\ \lambda_i^{-9} - 1\ ]\ \boldsymbol{n}_i \otimes \boldsymbol{N}_i - p\ \boldsymbol{F}^{-\text{t}}$.

Figure 15 summarizes the discovered weights for the principal stretch based network with L_p regularization for varying powers $p = [\ 0.5, 1.0, 2.0]$ and penalty parameters $\alpha = [\ 0.000, 0.001, 0.010, 0.100\ ]$. For all twelve combinations of the two hyperparameters, we perform a total of $n = 100$ training runs each, with varying initial conditions for the network weights $w_i = \{\ w_1, \ \dots\ , w_8\ \}$, such that each of the four models in Figure 14 is the result of one of the L_1 regularized training runs in the middle row. The colored boxes in Figure 15 indicate the relevance of the eight model terms, with their means and standard deviations. In contrast to the invariant based network in Section 4.1, the principal stretch based network consistently discovers similar terms across all three regularizations, with a clear preference for the dark blue $[\lambda_i^{-8} - 3]$ term. The fact that all networks robustly discover *similar* terms is a result of the *convex* nature of the underlying linear regression problem associated with the principal stretch based network and indicates the existence of a single *unique global minimum*. However, the $L_{0.5}$ and L_1 regularized networks gradually drop more non-zero terms as the penalty parameter increases, while the L_2 regularized network maintains all eight terms. As we had already anticipated from comparing Figures 3 and 5, the functional base of the principal stretch based network is more collinear than the base of the invariant based network, which result in a more gradual shift of the active weights, from w_1 towards w_8, as the penalty parameter increases. The fact that all three regularizations converge to the boundary of our domain, the dark blue $[\ \lambda_i^{-8} - 3\ ]$ term with the minimum exponent of minus eight, suggests that the true best fit might lay outside the current parameter range, with even smaller exponents. This agrees well with previous studies[50,62] that have discovered one-term Ogden models with exponents of $[\lambda_i^{-18} - 3]$ and $[\lambda_i^{-19} - 3]$.

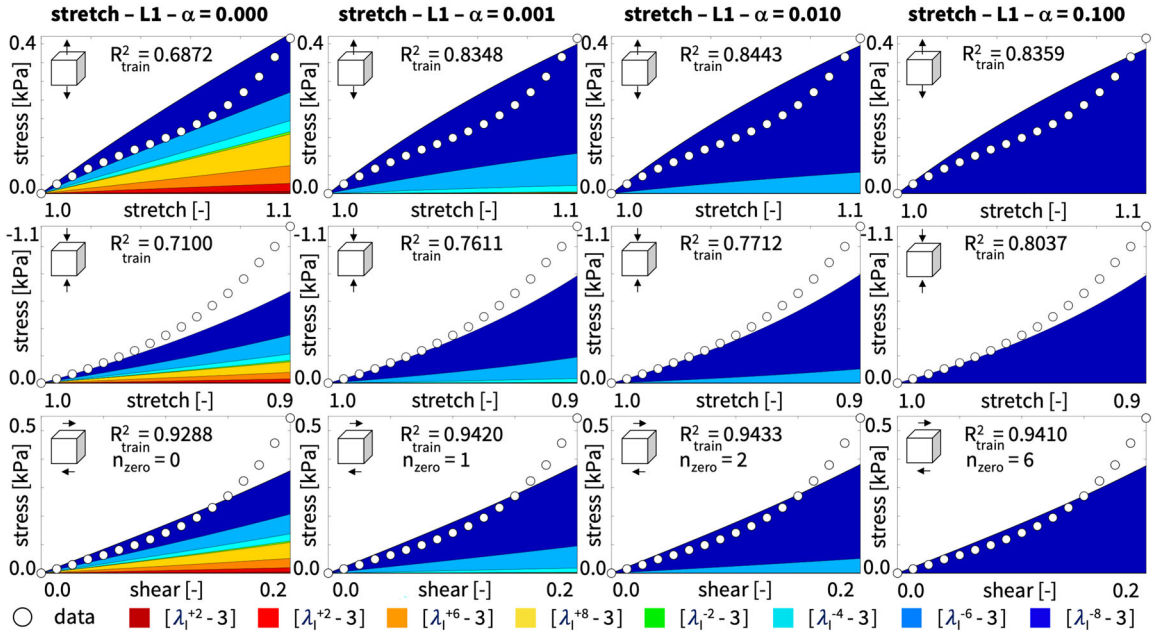

FIGURE 14 **Discovered models of L_1 regularized principal stretch based network.** Nominal stress as a function of stretch or shear strain for the principal stretch based neural network with L_1 regularization for varying penalty parameters $\alpha = [\ 0, 0.1, 0.001, 0.0001\ ]$, trained with human gray matter tension, compression, and shear data. Circles represent the experimental data. Color-coded regions represent the stress contributions of the eight model terms. Coefficients of determination R^2 indicate the goodness of fit.

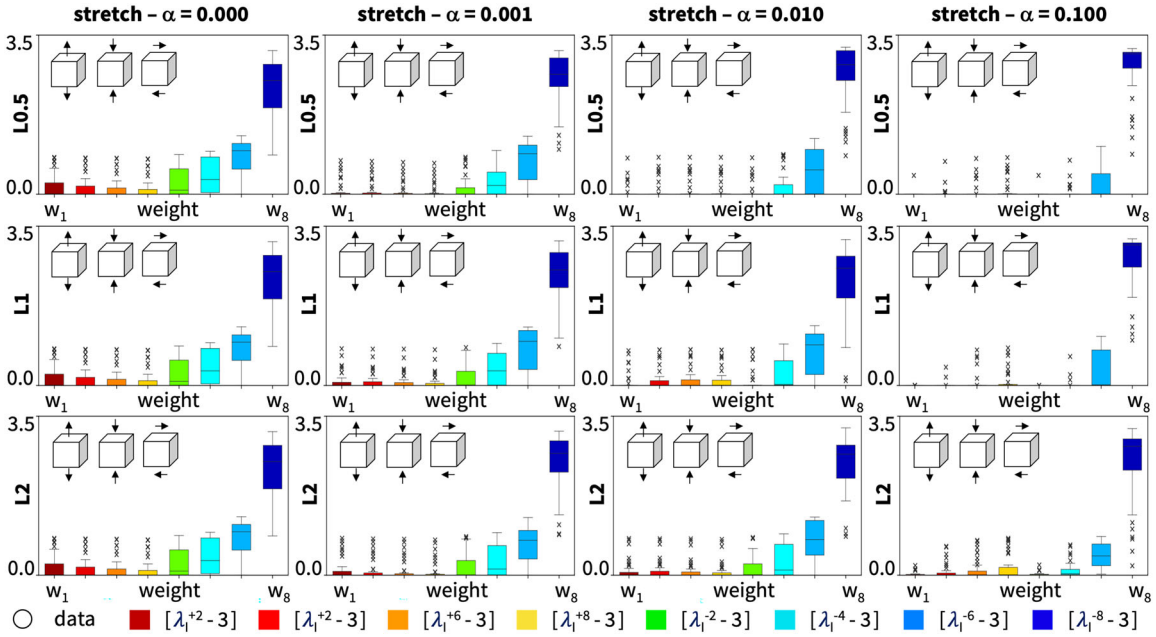

FIGURE 15 **Discovered models of L_p regularized principal stretch based network.** Distribution of discovered weights for the principal stretch based neural network with L_p regularization for varying powers $p = [\ 0.5, 1.0, 2.0]$ and penalty parameters $\alpha = [\ 0.000, 0.001, 0.010, 0.100\ ]$. Colored boxes indicate the relevance of the eight model terms, with means and standard deviations from $n = 100$ realizations with varying initializations of the network weights.

Taken together, this example illustrates that model discovery with L_p regularization *generalizes* well to different network types, irrespective of whether the terms are invariant or principal stretch based. Comparing both network types reveals that the method is sensitive to the *nonlinear* versus *linear* nature of the underlying regression problem: the nonlinear invariant based network alternates between different dominant terms associated with multiple local minima, while the linear principal stretch based network consistently discovers similar terms associated with a single unique global minimum.

5 | CONCLUSION AND RECOMMENDATIONS

L_p regularization is a powerful technology to finetune the training process of a neural network. In automated model discovery, it provides the critical missing piece of the puzzle that enables a controlled down-selection of the discovered terms and focus on the most important features of the model while putting less emphasis on minor effects. By promoting sparsity of the parameter vector, L_p regularization inherently improves interpretability and provides valuable insights into the underlying nature of the data. Importantly, L_p regularization introduces two hyperparameters: the power p by which it penalizes the individual model parameters, and the penalty parameter α by which it scales the relative importance of the regularization loss in comparison to the neural network loss. Both parameters enable a precise control of model discovery from data and it is crucial to understand their mathematical subtleties, computational implications, and physical effects. Here we reviewed the mathematics and computation of the most common representatives of the L_p family, and demonstrated their features in terms of two classes of constitutive neural networks, invariant and principal stretch based, trained with both, synthetic and real data. Training with synthetic data proved to be robust and stable, and generally provides excellent metrics for quality control since we know the exact solution. However, it remains a toy problem that fails to reveal the true usefulness in practical real-world applications. Training with real data was algorithmically robust, but challenging, since we know nothing about the exact solution. Our study uses neural networks as a tool for linear and nonlinear regression. We acknowledge that our results can be interpreted and well understood without resorting neural networks and generalize naturally to other regression techniques including symbolic regression, genetic programming, or system identification.

For conciseness, we have limited the scope of the present review: first, we only considered small networks with no more than eight terms, but point out that the automated model discovery generalizes well to isotropic networks with 12 terms,[29] transversely isotropic networks with 16 terms,[64] two-fiber family networks with 16 terms,[11] and orthotropic networks with 32 terms. Second, we trained on all available data and did not investigate splitting the data into train and test sets, which we have done in our previous work.[29,50] Third, we did not explicitly study the effects of controlled noise, but point out that Figures 8–15 are all based on real experimental data with real natural noise. Fourth, we did not further explore hybrid top down approaches like SINDy,[20] since the nonlinear nature of our optimization problem does not guarantee that we easily find the global minimum,[47] from which we could initiate a sequential thresholded least squares down-selection; finally, we have not yet investigated the effects of L_p regularization on uncertainty quantification, something we are currently exploring in a separate Bayesian approach.

We would like to share the most important lessons we have learnt throughout this study:

Normalize first! We cannot overstate the importance of normalizing. Clearly, while normalizing is less of an issue in linear regression, it is critical in nonlinear regression. This holds for both the training data, illustrated in Figures 3 and 5, and the weights, illustrated in Figure 11. The loss function typically contains several terms of different magnitude that compete during minimization. It proves important to normalize by the number of data sets in each category, the magnitude of the tensile, compressive, and shear stresses, and the magnitude of the weights to balance the impact of the individual contributions.

L_0 regularization is the most honest member of the L_p family: L_0 regularization or subset selection is honest, transparent, and unbiased. Its penalty parameter α acts as a *direct switch* to select the desired number of terms. It is the *only* member of the L_p family that explicitly controls the balance between the number of terms and the goodness of fit, illustrated in Figure 11. While L_0 regularization across the entire network translates into an expensive *NP hard discrete combinatorial problem* of the order of 2^n, we recommend to begin any discovery by running an L_0 regularization for all

possible one- and two-term models to determine the best-in-class models of each category and identify the dominant terms, similar to Figures 12 and 13 and Table 1. Importantly, in nonlinear regression, the best-in-class n-term model may actually *not* be a subset of the best-in-class $(n+1)$-term model, and *successively removing terms* like in iterative pruning[51] or sequential thresholding least squares[20] might not be a viable solution. Instead, running L_0 regularization for all possible one- and two-term models provides a quick first insight into the nature and hierarchy of the best-in-class models.[22] From this initial first glimpse, we can proceed by *successively adding terms*. In addition, from the best-in-class one-term models, we can use the discovered weights w_{i,L_0} to *initialize* the weights for higher order runs and to *normalize* the weights in the regularization term, $\alpha \,||\, \boldsymbol{\theta} \,||_p^p = \sum_i^{n_{\text{para}}} |\, w_i / w_{i,L_0} \,|^p$.

L_1 regularization is powerful for subset selection, but needs large penalty parameters to be effective: L_1 regularization or lasso promotes sparsity by reducing a large subset of parameters exactly to zero. Notably, for all the examples in our study, this down-selection required quite large penalty parameters α—often on the order of one-to work effectively. This not only affects the magnitude of the discovered parameters, but often also the discovered model itself. For example, for $\alpha = 0.1$, the $n = 100$ independent realizations of the L_1 regularization in Figure 10 alternate between the green and turquoise one-term models, while the unbiased plain L_0 regularization in Figure 12 and Table 1 ranks these two models clearly behind the blue and dark blue one-term models. To identify regularization-induced bias, we recommend to always compare the results of the L_1 regularization against the best-in-class low-term models of the L_0 regularization. This comparison is simple and inexpensive, and provides valuable insights into the magnitude of selection bias and the aggressiveness of the L_1 regularization.

$L_{0.5}$ regularization promotes sparsity for small penalty parameters, but suffers from multiple local minima: $L_{0.5}$ regularization addresses the shortcomings of the classical L_1 regularization by down-selecting more aggressively, requiring smaller penalty parameters, and introducing less bias. While $L_{0.5}$ regularization works well in practice, it is computationally challenging. Its non-convexity introduces multiple local minima, indicated through the first rows in Figures 6–8, and through the green and turquoise one-term models in Figure 10, and the blue and dark blue one-term models in Figure 15. To avoid getting stuck in a local minimum, we highly recommend exploring different initialization strategies for the network weights. Specifically, we were able to robustly identify multiple local minima by initializing the weights with the L_0 regularized weights w_{i,L_0}. Alternatively, we could gradually ramp up the effect of regularization by starting with a penalty parameter $\alpha = 0$ and smoothly increase it to a desired strength, essentially by moving from left to right in Figure 7. For quality control, we recommend comparing the remaining loss of each converged run against the remaining L_0 regularized baseline loss as reported in Table 1.

L_2 regularization promotes stability, but is not suited for subset selection: L_2 regularization, L by design, is not suited to reduce a subset of terms exactly to zero. Instead, it maintains all terms as indicated in the bottom rows of Figures 10 and 15, each for $n = 100$ runs. From Figures 6–8 we conclude that, for increasing penalty parameters α, L_2 regularization reduces outliers by *first* bringing the weights closer together and *then* collectively reducing them toward zero. Clearly, L_2 regularization improves convexity, which makes model discovery more robust and more stable. However, it not only fails to down-select the number of terms, but also strongly biases the solution away from the minimum of the pure network loss towards the minimum of the regularization loss. We do *not* recommend using L_2 regularization, or any other member of the L_p regularization family with powers larger than one, $p > 1$, to increase sparsity and improve interpretability in model discovery. Table 2 provides a side-by-side comparison of the L_p regularizations we explored throughout this study along with their advantages, disadvantages, and references.

Densifying instead of sparsifying: The *closure problem* is a common challenge in both fluid and solid mechanics. It refers to the difficulty of fully specifying the constitutive equations that relate stresses and strains and characterize the material behavior. In fluid mechanics, the closure problem is closely related to turbulence modeling, where it approximates intricate interactions between different scales, and can be well represented through *polynomials*.[20] In solid mechanics, the closure problem characterizes complex material behaviors at the microscopic scale, and is traditionally often represented through a combination of *polynomials*,[25] *exponentials*,[69,70] *logarithms*,[71] and *powers*.[57,66] In the context of model discovery, assuming perfect data, polynomial models translate into a convex linear optimization problem with a single unique global minimum, while exponential, logarithmic, or power models translate into a non-convex nonlinear optimization problem with possibly multiple local minima. For convex discovery problems with a unique global minimum, inducing sparsity has been well established through a top down approach in which we first calculate a dense

TABLE 2 L_p regularization.

	Algorithm	Regularization	Advantages	Disadvantages
L_0	**Subset selection**	$\alpha \, \|\|\theta\|\|_0$ $\|\|\theta \|\|_0 = \sum_i I(w_i \neq 0)$	• Penalizes number of non-zero terms • Term count is inherently unbiased • Conceptually simple and honest • *Promotes sparsity* • Improves interpretability • Valuable insight for one or two terms	• Solves *discrete combinatorial problem* • Results in *NP hard problem* • Computationally *expensive*, $\mathcal{O}(2^n)$ • But manageable for one term, $\mathcal{O}(n)$ • And manageable for two terms, $\mathcal{O}(n^2)$
$L_{0.5}$	Compromise between L_0 and L_1	$\alpha \, \|\|\theta\|\|_{0.5}^{0.5}$ $\|\|\theta \|\|_{0.5}^{0.5} = \sum_i \sqrt{\|w_i\|}$	• Improved efficiency compared to L_0 • Improved sparsity compared to L_1 • Reduces some parameters exactly to zero • Works even for smaller α and less bias	• Non-convex, multiple local minima • Increased computational complexity
L_1	**Lasso** least absolute shrinkage and selection operator	$\alpha \, \|\|\theta\|\|_1$ $\|\|\theta \|\|_1 = \sum_i \|w_i\|$	• Weighs all components equally • Less sensitive to outliers than L_2 • Reduces some parameters exactly to zero • *Promotes sparsity* • Improves interpretability	• Not strictly convex, *local minima* • Emphasizes selective effects • Introduces bias, *inaccurate for large* α
$L_{1/2}$	**Elastic net** compromise between L_1 and L_2	$\alpha_1 \, \|\|\theta\|\|_1 + \alpha_2 \, \|\|\theta\|\|_2^2$ $\|\|\theta \|\|_1 = \sum_i \|w_i\|$ $\|\|\theta \|\|_2^2 = \sum_i \|w_i\|^2$	• Improved stability compared to L_1 • Improved sparsity compared to L_2	• Increased computational complexity
L_2	**Ridge regression**	$\alpha_2 \, \|\|\theta\|\|_2^2$ $\|\|\theta \|\|_2^2 = \sum_i \|w_i\|^2$	• Uses components squared • Reduces outliers, *improves predictability* • Increases robustness • Promotes stability	• Introduces bias • Moves parameters towards each other • Reduces but maintains *all* parameters • Does *not* promote sparsity

Note: Comparison of special cases, advantages, disadvantages, and references.

parameter vector at the global minimum, and then *sparsify the parameter vector* by sequentially thresholding and removing the least relevant terms.[20,25,27,47] For non-convex discovery problems with multiple local minima, this approach is infeasible since different initial conditions may result in different solutions with non-unique parameter vectors.[67] Instead of trying to sparsify a dense parameter vector, we recommend to gradually *densify the parameter vector* from scratch. This bottom up approach iteratively solves the discrete combinatorial problem and densifies the parameter vector by sequentially adding the most relevant terms.[72] Importantly, instead of solving the NP hard discrete combinatorial problem associated with a complete L_0 regularization that screens all possible combinations of terms at $\mathcal{O}(2^n)$, we recommend to gradually add terms, starting with the best-in-class one-term model at $\mathcal{O}(n)$, adding a second term at $\mathcal{O}(n)$, and repeating addition until the incremental improvement of the overall loss function meets a user-defined convergence criterion. At most, this algorithm involves $\mathcal{O}(n^2)$ evaluations of the loss function to land on a fully populated dense parameter vector. Importantly, for non-convex model discovery problems, this algorithm—while cost effective and well-rationalized—is not guaranteed to converge to the global minimum. Instead of successively adding up to n terms, for practical purposes, it is often sufficient to limit the number of desirable terms to one, two, three or four, and identify the *best-in-class model* of each class, which requires a discrete comparison of $(\,8!/(n!(8-n)!)\,)$ discrete models, in our case 8, 28, 56, or 70.[22] Out of all possible discovery algorithms, this is the most honest, unbiased, and transparent approach.

Taken together, our study suggests that L_p regularized constitutive neural networks are a powerful technology for automated model discovery that allows us to identify interpretable constitutive models from data. We anticipate that our results generalize to L_p regularization for model discovery with other techniques such as symbolic regression or system

identification, and, more broadly, to model discovery in other fields such as biology, chemistry, or medicine. The ability to discover new knowledge from data could have tremendous applications in generative material design where it could shape the path to manipulate matter, alter properties of existing materials, and discover new materials with targeted properties.

AUTHOR CONTRIBUTIONS

JAMC: Method development; simulation; data analysis; result interpretation; manuscript writing. **SRSP**: Method development; simulation; data analysis; result interpretation; manuscript writing. **KL**: Study design; method development; simulation; data analysis; result interpretation; manuscript writing. **EK**: Study design; method development; simulation; data analysis; result interpretation; manuscript writing.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation Graduate Research Fellowship to Jeremy McCulloch and Skyler St. Pierre, by the DAAD Fellowship to Kevin Linka, and by the NSF CMMI Award 2320933 *Automated Model Discovery for Soft Matter* to Ellen Kuhl.

CONFLICT OF INTEREST STATEMENT

The authors declare no potential conflict of interests.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in the Living Matter Lab GitHub repository at https://github.com/LivingMatterLab/CANN, References 29 and 50.

ORCID

Skyler R. St. Pierre https://orcid.org/0000-0001-9774-8709
Kevin Linka https://orcid.org/0000-0002-1239-4778
Ellen Kuhl https://orcid.org/0000-0002-6283-935X

REFERENCES

1. Alber M, Buganza Tepole A, Cannon W, et al. Integrating machine learning and multiscale modeling: perspectives, challenges, and opportunities in the biological, biomedical, and behavioral sciences. *NPJ Digit Med*. 2019;2:115.
2. Brunton SL, Kutz JN. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. 1st ed. Cambridge University Press; 2019.
3. As'ad F, Avery P, Farhat C. A mechanics-informed artificial neural network approach in data-driven constitutive modeling. *Int J Numer Methods Eng*. 2022;123:2738-2759.
4. Fuhg JN, Bouklas N, Jones RE. Learning hyperelastic anisotropy from data via a tensor basis neural network. *J Mech Phys Solids*. 2022;168:105022.
5. Ghaderi A, Morovati V, Dargazany R. A physics-informed assembly for feed-forward neural network engines to predict inelasticity in cross-linked polymers. *Polymers*. 2020;12:2628.
6. Holthusen H, Lamm L, Brepols T, Reese S, Kuhl E. Theory and implementation of inelastic constitutive artificial neural networks; 2023. doi:10.48550/arXiv.2311.06380
7. Holzapfel GA, Linka K, Sherifova S, Cyron C. Predictive constitutive modelling of arteries by deep learning. *J R Soc Interface*. 2021;18:20210411.
8. Huang DZ, Xu K, Farhat C, Darve E. Learning constitutive relations from indirect observations using deep neural networks. *J Comput Phys*. 2020;416:109491.
9. Klein DK, Fernandez M, Martin RJ, Neff P, Weeger O. Polyconvex anisotropic hyperelasticity with neural networks. *J Mech Phys Solids*. 2022;159:105703.
10. Masi F, Stefanou I, Vannucci P, Maffi-Berthier V. Thermodynamics-based artificial neural networks for constitutive modeling. *J Mech Phys Solids*. 2021;147:04277.
11. Peirlinck M, Linka K, Hurtado JA, Kuhl E. On automated model discovery and a universal material subroutine for hyperelastic materials. *Comput Methods Appl Mech Eng*. 2024;418:116534.
12. Tac V, Sahli Costabal F, Buganza TA. Data-driven tissue mechanics with polyconvex neural ordinary differential equations. *Comput Methods Appl Mech Eng*. 2022;398:115248.
13. Tac V, Linka K, Sahli Costabal F, Kuhl E, Buganza TA. Benchmarking physics-informed frameworks for data-driven hyperelasticity. *Comput Mech*. 2023;73:49-65.

14. Wang LM, Linka K, Kuhl E. Automated model discovery for muscle using constitutive recurrent neural networks. *J Mech Behav Biomed Mater*. 2023;145:106021.
15. Linka K, Hillgartner M, Abdolazizi KP, Aydin RC, Itskov M, Cyron CJ. Constitutive artificial neural networks: a fast and general approach to predictive data-driven constitutive modeling by deep learning. *J Comput Phys*. 2021;429:110010.
16. Ghaboussi J, Garrett JH, Wu X. Knowledge-based modeling of material behavior with neural networks. *J Eng Mech*. 1991;117:132-153.
17. Linka K, Kuhl E. A new family of constitutive artificial neural networks towards automated model discovery. *Comput Methods Appl Mech Eng*. 2023;403:115731.
18. Bongard J, Lipson H. Automated reverse engineering of nonlinear dynamical systems. *Proc Natl Acad Sci USA*. 2007;104:9943-9948.
19. Schmidt M, Lipson H. Distilling free-form natural laws from experimental data. *Science*. 2009;324:81-85.
20. Brunton SL, Proctor JP, Kutz JN. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc Natl Acad Sci USA*. 2016;113:3932-3937.
21. Peng GCY, Alber M, Buganza Tepole A, et al. Multiscale modeling meets machine learning: what can we learn? *Arch Comput Methods Eng*. 2021;28:1017-1037.
22. Peirlinck M, Linka K, Hurtado JA, Holzapfel GA, Kuhl E. Democratizing biomedical simulation through automated model discovery and a universal material subroutine. *bioRxiv*, 2023. doi:10.1101/2023.12.06.570487
23. Simo JC, Taylor RL. Consistent tangent operators for rate-independent elastoplasticity. *Comput Methods Appl Mech Eng*. 1985;48:101-118.
24. Zienkiewicz OC, Taylor RL. *The Finite Element Method*. 4th ed. McGraw Hill College; 1987.
25. Flaschel M, Kumar S, De Lorenzis L. Unsupervised discovery of interpretable hyperelastic constitutive laws. *Comput Methods Appl Mech Eng*. 2021;381:113852.
26. Abdusalamov R, Hillgartner M, Itskov M. Automatic generation of interpretable hyperelastic models by symbolic regression. *Int J Numer Methods Eng*. 2023;124:2093-2104.
27. Wang Z, Estrada JB, Arruda EM, Garikipati K. Inference of deformation mechanisms and constitutive response of soft material surrogates of biological tissue by full-field characterization and data-driven variational system identification. *J Mech Phys Solids*. 2021;153:104474.
28. Flaschel M, Kumar S, De Lorenzis L. Automated discovery of generalized standard material models with EUCLID. *Comput Methods Appl Mech Eng*. 2023;405:115867.
29. Linka K, St Pierre SR, Kuhl E. Automated model discovery for human brain using constitutive artificial neural networks. *Acta Biomater*. 2023;160:134-151.
30. Frank IE, Friedman JH. A statistical view of some chemometrics regression tools. *Dent Tech*. 1993;35:109-135.
31. Hoerl AE, Kennard RW. Ridge regression: biased estimation for nonorthogonal problems. *Dent Tech*. 1970;12:55-67.
32. Santosa F, Symes WW. Linear inversion of band-limited reflection seismograms. *SIAM J Sci Stat Comput*. 1986;7:1307-1330.
33. Tibshirani R. Regression shrinkage and selection via the lasso. *J R Stat Soc B Methodol*. 1996;58:267-288.
34. Hastie T, Tibshirani R, Friedman J. *The Elements of Statistical Learning*. 2nd ed. Springer; 2009.
35. James G, Witten D, Hastie T, Tibshirani R. *An Introduction to Statistical Learning*. 2nd ed. Springer; 2013.
36. Koza JR. *Genetic Programming: on the Programming of Computers by Means of Natural Selection*. 1st ed. The MIT Press; 1992.
37. Seber GAF, Lee AJ. *Linear Regression Analysis*. 1st ed. John Wiley & Sons; 1977.
38. Antman SS. *Nonlinear Problems of Elasticity*. 2nd ed. Springer; 2005.
39. Fuhg JN, Bouklas N. On physics-informed data-driven isotropic and anisotropic constitutive models through probabilistic machine learning and space-filling sampling. *Comput Methods Appl Mech Eng*. 2022;394:114915.
40. Hartmann S. Parameter estimation of hyperelastic relations of generalized polynomial-type with constraint conditions. *Int J Solids Struct*. 2001;38:7999-8018.
41. Linden L, Klein DK, Kalinka KA, Brummund J, Weeger O, Kästner M. Neural networks meet hyperelasticity: a guide to enforcing physics. *J Mech Phys Solids*. 2023;179:105363.
42. Tac V, Rausch MK, Sahli Costabal F, Buganza TA. Data-driven anisotropic finite viscoelasticity using neural ordinary differential equations. *Comput Methods Appl Mech Eng*. 2023;411:116046.
43. Truesdell C, Noll W. Non-linear field theories of mechanics. In: Flügge S, ed. *Encyclopedia of Physics*. Vol III/3. Spinger; 1965.
44. Seber GAF, Wild CJ. *Nonlinear Regression*. 1st ed. John Wiley & Sons; 1989.
45. Bates DM, Watts DG. *Nonlinear Regression Analysis and its Applications*. 1st ed. John Wiley & Sons; 1988.
46. Champion K, Lusch B, Kutz N, Brunton SL. Data-driven discovery of coordinates and governing equations. *Proc Natl Acad Sci USA*. 2019;116:22445-22451.
47. Zhang L, Schaeffer H. On the convergence of the SINDy algorithm. *Multiscale Model Simul*. 2019;17:948-972.
48. Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators. *Neural Netw*. 1989;2:359-366.
49. Goodfellow I, Bengio Y, Courville A. *Deep Learning*. 1st ed. The MIT Press; 2016.
50. St Pierre SR, Linka K, Kuhl E. Principal-stretch-based constitutive neural networks autonomously discover a subclass of Ogden models for human brain tissue. *Brain Multiphys*. 2023;4:100066.
51. Han S, Pool J, Tran J, Dally WJ. Learning both weights and connections for efficient neural networks. *Proceedings of the 28th International Conference on Neural Information Processing Systems*. Vol 1. MIT Press; 2015:1135-1143.
52. LeCun Y, Denker JS, Solla SA. Optimal brain damage. *Advances in Neural Information Processing Systems 2*. Morgan-Kaufmann; 1989.
53. Fu WJ. The bridge versus the lasso. *J Comput Graph Stat*. 1998;7:397-416.
54. Zou H, Hastie T. Regularization and variable selection via the elastic net. *J R Stat Soc Series B Stat Methodol*. 2005;67:301-320.
55. Friedman JH. Sparse regression in and classification. *Int J Forecast*. 2012;28:722-738.

56. Holzapfel GA. *Nonlinear Solid Mechanics: A Continuum Approach to Engineering*. John Wiley & Sons; 2000.
57. Ogden RW. Large deformation isotropic elasticity—on the correlation of theory and experiment for incompressible rubberlike solids. *Proc R Soc Lond A Math Phys Sci*. 1972;326:565-584.
58. Blatz PJ, Ko WL. Application of finite elastic theory to the deformation of rubbery materials. *Trans Soc Rheol*. 1962;6:223-251.
59. Mooney M. A theory of large elastic deformations. *J Appl Phys*. 1940;11:582-590.
60. Rivlin RS. Large elastic deformations of isotropic materials. IV. Further developments of the general theory. *Philos Trans R Soc Lond A*. 1948;241:379-397.
61. Treloar LRG. Stresses and birefringence in rubber subjected to general homogeneous strain. *Proc Phys Soc*. 1948;60:135-144.
62. Budday S, Sommer G, Birkl C, et al. Mechanical characterization of human brain tissue. *Acta Biomater*. 2017;48:319-340.
63. Budday S, Ovaert TC, Holzapfel GA, Steinmann P, Kuhl E. Fifty shades of brain: a review on the material testing and modeling of brain tissue. *Arch Comput Methods Eng*. 2020;27:1187-1230.
64. Linka K, Buganza Tepole A, Holzapfel GA, Kuhl E. Automated model discovery for skin: discovering the best model, data, and experiment. *Comput Methods Appl Mech Eng*. 2023;410:116007.
65. St Pierre SR, Rajasekharan D, Darwin EC, Linka K, Levenston ME, Kuhl E. Discovering the mechanics of artificial and real meat. *Comput Methods Appl Mech Eng*. 2023;415:116236.
66. Valanis K, Landel RF. The strain-energy function of a hyperelastic material in terms of the extension ratios. *J Appl Phys*. 1967;38:2997-3002.
67. Ogden RW, Saccomandi G, Sgura I. Fitting hyperelastic models to experimental data. *Comput Mech*. 2004;34:484-502.
68. Kingma DP, Ba J. Adam: a method for stochastic optimization; 2014. doi:10.48550/arXiv.1412.6980
69. Demiray H. A note on the elasticity of soft biological tissues. *J Biomech*. 1972;5:309-311.
70. Holzapfel GA, Gasser TC, Ogden RW. A new constitutive framework for arterial wall mechanics and comparative study of material models. *J Elast*. 2000;61:1-48.
71. Gent A. A new constitutive relation for rubber. *Rubber Chem Technol*. 1996;69:59-61.
72. Nikolov DP, Srivastava S, Abeid BA, et al. Ogden material calibration via magnetic resonance cartography, parameter sensitivity and variational system identification. *Philos Trans A Math Phys Eng Sci*. 2022;380:20210324.

How to cite this article: McCulloch JA, St. Pierre SR, Linka K, Kuhl E. On sparse regression, L_p-regularization, and automated model discovery. *Int J Numer Methods Eng*. 2024;125(14):e7481. doi: 10.1002/nme.7481