Choquet Integral Ridge Regression

Siva K. Kakula[a], Anthony J. Pinar[b], Timothy C. Havens[ab]
[a]College of Computing
[b]Department of Electrical and Computer Engineering
Michigan Technological University
Houghton, MI 49931, USA
e-mail: {skakula, ajpinar, thavens}@mtu.edu

Derek T. Anderson
Department of Electrical Engineering and Computer Science
University of Missouri
Columbia, MO, 65211, USA
e-mail: andersondt@missouri.edu

***Abstract*—The Choquet integral (ChI) is an aggregation function that is defined with respect to a fuzzy measure (FM). Many ChI-based decision aggregation methods have been proposed to learn the underlying FM. However, FM's boundary and monotonicity constraints have limited the applicability of such methods to decision-level fusion. In a recent work, we removed the constraints on FM to develop a regression model based on the ChI. Our model has a generalized bias that enables capability beyond previously proposed ChI regression approaches. We also developed an approach for learning the parameters of the ChI regression from training data. In this paper, we develop a method to apply ℓ_2-regularization on our training algorithm. In our experiments on real-world benchmark data sets, ridge regularized-ChI regression has outperformed the unregularized version in 22 out of 30 (73%) experiments. Also, when compared with several competing regression methods, results show that our approach has superior performance.**

***Keywords*—regression, Choquet integral, fuzzy integral, fuzzy measure, regularization, machine learning**

I. Introduction

Regression approaches typically seek a function, $f(\cdot)$, that can transform or map an independent variable, $\mathbf{x}$, to a dependent variable, y, given a training set of d-dimensional independent variables, $\mathbf{x} \in \mathcal{R}^d$, and 1-dimensional dependent variables, $y \in \mathcal{R}$. The relationship between $\mathbf{x}$ and y is a parameterized model (or function), such that

$$Y \approx r(\mathbf{x}, \alpha),$$

where α is the set of learned parameters of the regression function r. A linear model is a common choice for this function, $r(\mathbf{x}, \alpha) = \mathbf{w}^T\mathbf{x} + \beta$, where $\mathbf{w} \in \mathcal{R}^d$ and β is the bias. Using basis functions, we can extend this linear model to learn non-linear relations, $r(\mathbf{x}, \alpha) = \mathbf{w}^T\phi(\mathbf{x}) + \beta$, where ϕ is a set of basis functions. Examples for basis functions include quadratic, polynomial, and radial basis functions. Basis functions typically project the input data $\mathbf{x}$ into a higher-dimensional space, which allows the regression function to learn more complex non-linear relations (in the native space). However, inclusion of basis functions often eliminates the interpretability of the results. This is because, without basis functions, the learned parameter vector $\mathbf{w}$ directly indicates how each variable in the input $\mathbf{x}$ affects the output, but when basis functions are included, the parameter vector $\mathbf{w}$ is not as interpretable.

The regression parameters α are typically trained using a set of training data pairs $(X, Y) = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$. The training process optimizes the regression parameters α with respect to an error function, usually squared-error:

$$\alpha^* = \arg\min_{\alpha} \sum_{i=1}^{n} \left(r(\mathbf{x}_i, \alpha) - y_i\right)^2.$$

This is the well-known least-squares problem. For more extensive details on regression, in general, we suggest [1].

In our recent previous work [2], we proposed a regression model based on the *Choquet integral* (ChI) with respect to a *bounded capacity* (BC) [3], where we demonstrated that ChI, in combination with a *fuzzy measure* (FM)[1] can produce a non-linear aggregation method, which essentially is a compressed parameterization of a set of linear convex sums, one for each sort order of the input. Since the number of parameters in an FM scales as 2^d, where d is the number of variables in the input, training the ChI-FM regression model on a high-dimensional input data requires a large training data set with enough *rank-independent* observations. However, many real-world data sets often do not contain enough *rank-independent* observations; training on such data sets typically results in an overfit model that does not generalize well on the unseen data. In this work, we build on the prior work by addressing the overfitting problem of our ChI-FM regression method with the introduction of ℓ_2-regularization in the training process.

Several previous works have explored regression using ChIs [4–6]. Some works explored certain types of FMs, such as Sugeno's λ-measure, P-measures, or L-measures, e.g., [5]. Different from these works, the learning approach of our method enables us to learn *any* bounded capacity, which is more general than a parameterized FM. Grabisch [4] proposed a method to learn an FM using training data; however, since this approach is limited to FMs, it has limited applicability. In addition, this approach used a single bias value model (similar to our CIR(1) model proposed here), which limits the flexibility of the method. Our method learns a more flexible bias model (up to one bias for each possible sort). ChI was also applied to logistic regression by Tehrani et al. [6]; this approach also included a single bias value (like our CIR(1) model). Our method could be used to extend this logistic

[1]A fuzzy measure is a bounded monotonic capacity on $[0, 1]$; see Sec. II.

978-1-7281-6932-3/20/$31.00 ©2020 IEEE

TABLE I: Notations and Acronyms

Acronym	Description	Notation
	input data or evidence on X	$h, \mathbf{h}$
	descending sorting function	π
BC	(non-monotonic) bounded capacity	f
FM	(monotonic) fuzzy measure	g
FI	fuzzy integral	
ChI	Choquet fuzzy integral	$C_g(h)$
CIR	Choquet integral regression	$C_f(h)$
	training data pairs: (input, output)	$(\mathbf{h}_i, y_i)$

regression approach. Recently, Du and Zare [7] proposed a multiple instance learning ChI regression, though a bias model was not included in this regression approach.

Our method [2] was a significant extension of these prior works as it used a bounded capacity and a bias model that allowed encoding of a linear model for each possible sort in the input data. In [2] our experimental results showed better performance than comparable methods. In this work, we further improve our method by applying ℓ_2-regularization. Our experimental results show that the application of ℓ_2-regularization has consistently improved the performance as compared to the unregularized version.

ChIs with respect to non-monotonic measures have been discussed in [8–11]. These works provide a good basis for the development and application of ChIs with respect to non-monotonic measures. Our work here is a significant extension to these works to enable a generalized ChI-based regression model along with regularization.

The remainder of this paper is organized as follows. Section II presents the background on ChIs and BCs, then Section III discusses our ChI regression (CIR) method. We introduce ridge regression on CIR in Section IV. We compare the normal as well as the regularized ChI regression with several other regression models in Section V. Section VI summarizes. The acronyms and notations used in this paper are presented in the Table I.

II. Background

A. Fuzzy measures

We consider a measurable space as the tuple (X, Ω), where X is a set and Ω is a σ-algebra or set of subsets of X such that

P1. $X \in \Omega$;
P2. For $A \subseteq X$, if $A \in \Omega$, then $A^c \in \Omega$;
P3. If $\forall A_i \in \Omega$, then $\bigcup_{i=1}^{\infty} A_i \in \Omega$.

An FM is a set-valued function, $g : \Omega \to [0, 1]$, with the following properties:

P4. (Boundary conditions) $g(\emptyset) = 0$ and $g(X) = 1$;
P5. (Monotonicity) If $A, B \in \Omega$ and $A \subseteq B$, $g(A) \leq g(B)$.

There is an additional property that guarantees continuity for the case where Ω is an infinite set, however, in practice and in this paper, Ω is finite and thus this property is unnecessary. FMs provide us with a convenient way to quantify the worth of combinations of sources, and *fuzzy integrals* (FI) can be applied over FMs to aggregate the information from these sources. The FM values of the singletons, $g(\{x_i\}) = g^i$ are commonly called the *densities*.

An FM that obeys all the above properties, when used for aggregation with ChI, results in convex sums of input variables that are bounded between maximum and minimum values of the input variables. To enable a continuous unbounded aggregation of outputs, we will relax the properties of the FM to a BC.

Definition 1. A BC is a set-valued function $f : \Omega \to \mathcal{R}$, with the boundary property $f(\emptyset) = 0$.[2]

B. Fuzzy integrals

There are many forms of the FI; see [3] for a detailed discussion. Several previous works [12–17] have explored FIs as a tool for evidence fusion. The FM provides the expected worth of each subset of the sources, and the FIs use this to combine information from the sources while accounting for both the support of the evidence as well as the expected worth. In this paper, we focus on the ChI proposed by Murofushi and Sugeno [18, 19]. Let $h : X \to \mathcal{R}$ be a real-valued function that represents the evidence or support of a particular hypothesis.[3] In the remainder of this paper, we will shorten $h(x_i)$ to h_i and can thus present a collection of inputs in vector form, $\mathbf{h} = (h_1, h_2, \ldots, h_d)^T$.

The discrete (finite Ω) ChI is defined as

$$\int_C \mathbf{h} \circ g = C_g(\mathbf{h}) = \sum_{i=1}^{d} h_{\pi(i)} \left[g(\Pi_i) - g(\Pi_{i-1}) \right], \quad (1a)$$

$$= \gamma_\pi^T \mathbf{h}_\pi, \quad (1b)$$

where π is a permutation of X, such that $h_{\pi(1)} \geq h_{\pi(2)} \geq \ldots \geq h_{\pi(d)}$, $\Pi_i = \{x_{\pi(1)}, \ldots, x_{\pi(i)}\}$, and $g(\Pi_0) = 0$ [20, 21]. In (1b), we have simply reformulated (1a) as the dot-product of the vectors γ_π and $\mathbf{h}_\pi$, where

$$\mathbf{h}_\pi = \left(h_{\pi(1)},\ h_{\pi(2)},\ \ldots,\ h_{\pi(d)} \right)^T, \quad (2)$$

$$\gamma_\pi = \left(g(\Pi_1),\ (g(\Pi_2) - g(\Pi_1)),\ \ldots,\ (1 - g(\Pi_{d-1})) \right)^T. \quad (3)$$

The key insight from (1b) is that the ChI with respect to the FM is essentially a collection of $d!$ linear-order statistics on h, one for each possible sort order of the evidence h.[4] The elements of γ_π are simply the weights of each evidence value, as represented by the gain in the FM up through the lattice. More detail on the properties of fuzzy ChIs and fuzzy integrals in general can be found in [20–22].

[2] With regard to Choquet integral regression, described in Section III, this boundary condition on f is arbitrary. It can be shown that any constant bias applied to all values in f does not change the result of the regression. This boundary condition can help with optimization from a practical computing standpoint.

[3] Generally, when dealing with an information fusion problems it is convenient to have $h : X \to [0, 1]$, where each source is normalized to the unit-interval.

[4] It can be further specified to say that the ChI is a collection of $d!$ *ordered weighted averages* (OWA), as $\sum_i (\gamma_\pi)_i = 1$.

III. Choquet Integral Regression

A. CIR formulation

We can directly extend the ChI at (1) to *Choquet integral regression* (CIR) by integrating with respect to a BC f and adding a bias term,

$$\int_C \mathbf{h} \circ f = C_f(\mathbf{h}) = \beta_\pi + \sum_{i=1}^{d} h_{\pi(i)} \left[f(\Pi_i) - f(\Pi_{i-1}) \right], \tag{4a}$$

$$= \beta_\pi + \rho_\pi^T \mathbf{h}_\pi, \tag{4b}$$

where π and Π are defined as in (1), $\mathbf{h}_\pi$ is defined at (3), $\beta_\pi \in \mathcal{R}$ is a bias term[5], and

$$\rho_\pi = \Big((f(\Pi_1) - f(\emptyset)),\ (f(\Pi_2) - f(\Pi_1)),\ \ldots, \\ (f(X) - f(\Pi_{d-1})) \Big)^T. \tag{5}$$

In an FM g, $f(\emptyset)$ and $f(X)$ are assigned the static boundary values of 0 and 1, respectively. We remove these boundary constraints in (5) by including $f(\emptyset)$ and $f(X)$ in ρ_π; see γ_π at (3) for comparison.

How is the CIR formulation at (4) a regression? Consider the formulation of CIR at (4b)—with β_π as the learned bias and ρ_π as the learned weight vector, this clearly is in the form of linear regression. In addtion, since we removed the boundary constraints, the values in ρ_π can take any value in the set of reals, $\mathcal{R}$—see (5). This makes the CIR a compressed parameterization of a set of linear convex sums. In other words, CIR is a set of linear regressions, one for each of the $d!$ possible sorts of h. The ChI, since it is an aggregation operator, produces outputs that are bounded by the interval $[\min\{\mathbf{h}\}, \max\{\mathbf{h}\}]$; while the CIR, by allowing the learned parameters in ρ_π to take any value in $\mathcal{R}$, enables the mapping of inputs to anywhere in the set of reals, $C_f(h) \in \mathcal{R}$, and is therefore a regression operator.

B. CIR learning

Given a set of training data pairs $\{(\mathbf{h}_1, y_1), \ldots, (\mathbf{h}_n, y_n)\}$, $y_i \in \mathcal{R}, \forall i$, we would like to learn a prediction function o such that $o(\mathbf{h}_i) = y_i$. This is a standard regression problem. In our prior work [23, 24], we explored the approaches to train the ChI as a prediction function, by minimizing the *sum of squared error* (SSE) between the true output and the prediction for a given set of training data, i.e.,

$$g^* = \arg\min_g \left\{ \sum_{i=1}^{n} \left(C_g(\mathbf{h}_i) - y_i \right)^2 \right\}. \tag{6}$$

It can be shown that the solution to (6) is the *quadratic program* (QP)

$$\min_{\mathbf{u}_g} \mathbf{u}_g^T D \mathbf{u}_g + \mathbf{t}^T \mathbf{u}_g, \quad C\mathbf{u}_g \leq \mathbf{0}, \quad (\mathbf{0}, 1)^T \leq \mathbf{u}_g \leq \mathbf{1}, \tag{7}$$

where $\mathbf{u}_g$ is the lexographically-ordered FM g, i.e., $\mathbf{u}_g = (g(\{x_1\}), g(\{x_2\}), \ldots, g(\{x_1, x_2\}), g(\{x_1, x_3\}), \ldots, g(\{x_1, x_2, \ldots, x_d\}))$; the matrices D and $\mathbf{t}$ are composed of the training data $\mathbf{h}$ and $\mathbf{y}$; and the matrix C enforces the monotonicity property on the learned FM g. Our prior work [24] contains the details on the construction of these matrices and the implementation of the QP. Using this QP-based learning approach, we applied ChI on many sensor fusion problems; however, the output of ChI is limited to the interval between the maximum and the minimum of the inputs. Hence, we explore next, the process to learn the CIR, which does not have such limitations.

In this manuscript, we focus on the BC f and the CIR at (4). Therefore, we would like to solve the minimization problem

$$(f^*, \beta^*) = \arg\min_{f,\beta} \left\{ \sum_{i=1}^{n} (C_f(\mathbf{h}_i) - y_i)^2 \right\}. \tag{8}$$

First, we will rewrite (4) as

$$C_f(\mathbf{h}_i) = \beta_\pi + \sum_{j=0}^{d} f(\Pi_j) \left[(\mathbf{h}_i)_{\pi(j)} - (\mathbf{h}_i)_{\pi(j+1)} \right], \tag{9}$$

where we define $(\mathbf{h}_i)_{\pi(0)} = (\mathbf{h}_i)_{\pi(d+1)} = 0$, $\forall i$. Since $f(\Pi_0) = f(\emptyset) = 0$, the first summation term equals 0. However, if the user wants to set different boundary conditions, or none at all, this term can be adjusted accordingly. To continue, f is then lexographically ordered, i.e., $\mathbf{u}_f = (f(\emptyset), f(\{x_1\}), f(\{x_2\}), \ldots, f(\{x_1, x_2\}), f(\{x_1, x_3\}), \ldots, f(\{x_1, x_2, \ldots, x_d\}))$.[6] The SSE term (8) on expansion gives

$$E^2 = \sum_{i=1}^{n} (C_f(\mathbf{h}_i) - y_i)^2 = \sum_{i=1}^{n} (H_i^T \mathbf{u}_f + B_i^T \beta - y_i)^2, \tag{10}$$

where H_i is a $2^d \times 1$ vector that contains the difference terms in (9); B_i is a $b \times 1$ bit vector that chooses (or computes) the bias from the vector β for a given training data pair (more on that soon).

To transform (10) into a standard least-squares problem, we concatenate a variable vector $\mathbf{u} = (\mathbf{u}_f, \beta)^T$, and then build the vector

$$D_i = \left(H_i^T, B_i^T \right). \tag{11}$$

Thus, (10) becomes

$$E^2 = \sum_{i=1}^{n} (D_i \mathbf{u} - y_i)^2, \tag{12}$$

This has the standard least-squares solution of

$$\mathbf{u} = (D^T D)^{-1} D^T \mathbf{y}, \tag{13a}$$

$$D = \begin{bmatrix} D_1 \\ D_2 \\ \vdots \\ D_n \end{bmatrix}. \tag{13b}$$

[5] At this point in the manuscript, we choose to generalize the bias so that one could have up to $d!$ different bias terms, one for each sort; however, as we will explore in Section III-B, this may be computationally intractable—$d!$ can grow to be a large number—and, hence, we will also develop solutions that use fewer bias terms.

[6] One could choose any ordering of f and derive the QP matrices appropriately. For example, in our code library we use binary ordering, such that $\mathbf{u}_f = (f(\emptyset), f(\{x_1\}), f(\{x_2\}), f(\{x_1, x_2\}), f(\{x_3\}), f(\{x_1, x_3\}), \ldots, f(X))$.

This solution is satisfying as it further bolsters our claim that CIR is regression at its core. We used the least-squares solver in Matlab for the results presented in this paper. Problems such as singular matrices and underdetermined systems are automatically addressed by the solver. Note that the least-squares solution at (13) is a mathematically correct solution, but can be problematic due to its inverse; hence, least-square solvers are often more stable in practice.

We now describe the process to build the matrices H_i and B_i. The approach for a simple 3-input case is show in Example 1. It may be useful to follow along with this example as we describe the process in more detail.

The $2^d \times 1$ H_i matrix is designed to contain the $(d+1)$ difference terms, $\left[(\mathbf{h}_i)_{\pi(j)} - (\mathbf{h}_i)_{\pi(j+1)}\right]$, in (9). The rest of the elements of H_i are all zero. That is,

$$H_i = \begin{pmatrix} 0 - (\mathbf{h}_i)_{\pi(1)} \\ \vdots \\ 0 \\ \vdots \\ (\mathbf{h}_i)_{\pi(1)} - (\mathbf{h}_i)_{\pi(2)} \\ \vdots \\ 0 \\ \vdots \\ (\mathbf{h}_i)_{\pi(d)} - 0 \end{pmatrix}. \tag{14}$$

The B_i matrix varies based on the user's choice of the β vector construction. Though one could imagine numerous ways to learn the bias vector β, in this paper, we present three possible choices:

1) Use a single scalar β value; thus, $B_i = 1$, $\forall i$. This is simplest and least computationally expensive choice.
2) Pick the β value based on the first element in the sort order $\pi(1)$, i.e., β is only dependent on the input with greatest magnitude. Thus, B_i is $d \times 1$ and takes the form $[B_i]_{\pi(1)} = 1$, else $[B_i] = 0$.
3) The third method mimics the lattice of the BC and sets B_i according to the non-zero elements of H_i. That is,

$$[B_i]_j = \begin{cases} 1 & [H_i]_j > 0, \\ 0 & \text{else.} \end{cases} \tag{15}$$

In the third method, B_i is a $2^d \times 1$ matrix with $d+1$ entries set to 1. Thus, for each sort order, the bias value in the regression solution is the sum of $d+1$ elements of the β vector. In this way, the CIR can thus learn the β vector to produce a different bias for each sort order, but encoding this bias with only 2^d values (rather than $d!$). Table II outlines the three methods for learning the bias.

One could also imagine choosing a β value for each possible sort order; thus, B_i is $d! \times 1$ and has one entry that is set to 1 depending on the coding of the sort order. We view this choice as intractable in practice, as $d!$ can become very large, e.g., $10! = 3,628,800$.

TABLE II: Three Methods for Building Bias Vector β

Name	Description	B_i
1-bias	One bias term	$B_i = 1$, $\forall i$
d-bias	One term for each max-value in sort	$[B_i]_{\pi(1)} = 1$ else $[B_i] = 0$
2^d-bias	Computed bias for each possible sort	See (15)

Example 1. For this example, we will use a β vector that is $2^d \times 1$. Consider two training data pairs ($n = 2$) with three inputs ($d = 3$) as follows:

$$(\mathbf{h}_1, y_1) = ((1,2,4)^T, 8),$$
$$(\mathbf{h}_2, y_2) = ((4,-6,1)^T, 2).$$

The sort order for data pair 1 is $\Pi = (3,2,1)$, and the sort order for data pair 2 is $\Pi = (1,3,2)$. Thus, the QP sub-matrices for this example are

$$H_1 = (-4,0,0,2,0,0,1,1)^T,$$
$$H_2 = (-4,3,0,0,0,7,0,-6)^T,$$
$$B_1 = (1,0,0,1,0,0,1,1)^T,$$
$$B_2 = (1,1,0,0,0,1,0,1)^T.$$

For this example, the least-squares problem is underdetermined. Using Matlab's solver, the solution is $\mathbf{u}_f = (-2,0,0,0,0,-0.86,0,0)^T$, and $\beta = \mathbf{0}$.

We now extend CIR with regularization to account for overfitting in learning.

IV. CIR with Ridge Regularization

Recall the insight of (4) (or (1b)) where we observe that the ChI encodes $d!$ regression models (or linear-order statistics), one for each of the $d!$ possible sort orders. Let us enumerate these sorting orders as $\pi_1, \pi_2, ..., \pi_{d!}$, leading to the $d!$ regression models, $\rho_{\pi_1}, \rho_{\pi_2}, ..., \rho_{\pi_{d!}}$. We then modify the SSE cost function in (12) to include ℓ_2 (ridge) regularization terms for all $d!$ models, yielding

$$E_R^2 = \sum_{i=1}^{n} (D_i \mathbf{u} - y_i)^2 + \lambda \sum_{j=1}^{d!} \|\rho_{\pi_j}\|_2^2, \tag{16}$$

where λ is the regularization parameter defining the weight of the regularizer—a user-tuned quantity. To solve this, we must express the various regression models, ρ_{π_j}, in terms of the BC, $\mathbf{u}_f$. This is accomplished by defining a $d \times 2^d$ matrix $A_{\rho_{\pi_i}}$ to sift the regression model from $\mathbf{u}_f$, i.e.,

$$\rho_{\pi_i} = A_{\rho_{\pi_i}} \mathbf{u}_f. \tag{17}$$

Note the matrix $A_{\rho_{\pi_i}}$ is mostly zeros, and each row has at-most two non-zero elements (one $+1$ and one -1); this matrix-vector product produces the difference terms shown in (5). Also note that since the ridge regression term at (16) is applied to the product $A_{\rho_{\pi_i}} \mathbf{u}$, we can interpret the term as a Tikhonov regularizer and the matrix $A_{\rho_{\pi_i}}$ can be thought of as a Tikhonov matrix [25].

Each sort order will have its own $A_{\rho_{\pi_i}}$, thus there will be $d!$ unique sifting matrices. We now express (16) in terms of $\mathbf{u}_f$ as

$$E_R^2 = \sum_{i=1}^{n}(D_i\mathbf{u} - y_i)^2 + \lambda\sum_{j=1}^{d!}\rho_{\pi_j}^T\rho_{\pi_j} \tag{18a}$$

$$= \sum_{i=1}^{n}(D_i\mathbf{u} - y_i)^2 + \lambda\sum_{j=1}^{d!}\mathbf{u}_f^T A_{\rho_{\pi_j}}^T A_{\rho_{\pi_j}}\mathbf{u}_f \tag{18b}$$

$$= \sum_{i=1}^{n}(D_i\mathbf{u} - y_i)^2 + \lambda\mathbf{u}_f^T\hat{G}_\rho\mathbf{u}_f, \tag{18c}$$

where $\hat{G}_\rho = \sum_{j=1}^{d!}\left(A_{\rho_{\pi_j}}^T A_{\rho_{\pi_j}}\right)$. Note that $\hat{G}_\rho$ is a constant sparse matrix of size $2^d \times 2^d$ and only depends on d, thus it can be built offline. Finally, appending block matrices of zeros to $\hat{G}_\rho$ to make it compatible with the concatenated vector $\mathbf{u}$ allows the ridge cost function to be written as

$$E_R^2 = \sum_{i=1}^{n}(D_i\mathbf{u} - y_i)^2 + \lambda\mathbf{u}^T G_\rho\mathbf{u}, \tag{19}$$

where $G_\rho = \begin{pmatrix}\hat{G}_\rho & 0\\ 0 & 0\end{pmatrix}$. Unfolding the squared term in (19), setting the gradient with respect to $\mathbf{u}$ equal to zero, and solving for $\mathbf{u}$ gives the minimizer

$$\hat{\mathbf{u}}_R = \left(D^T D + \lambda G_\rho\right)^{-1} D^T\mathbf{y}. \tag{20}$$

This solution is satisfying since its form is very similar to the well-known ridge regression solution.[7] Note that if $G_\rho = I$, where I is the identity matrix, the solution matches that of ridge regression exactly. This is not surprising; because the fuzzy measure vector we are learning here has a different structure than the typical regression weight vector, the regularization matrix, G_ρ, must have a different structure than the identity matrix to compensate.

It is also interesting to note that since ρ_{π_i} is defined by a subset of $\mathbf{u}_f$, the objective function at (16) is essentially group lasso performed on $\mathbf{u}_f$ [26]. Furthermore, since each individual element of $\mathbf{u}_f$ appears in more than one ρ_{π_i}, it is the more general case of group lasso with overlapping groups [27, 28].

V. Experiments

We evaluated the performance of our CIR methods and the impact of ℓ_2-regularization using real world data sets from the UCI Machine Learning repository [29]. In addition, we compared the performance with several other regression methods; Table III presents the details on methods used in our experiments. The methods *Interactions*, *PureQuadratic*, and *Quadratic* use basis functions (indicated by $\phi(\mathbf{h})$ in Table III) to project the input data into a higher-dimensional space and thereby induce non-linearity into the regression solutions. Note that these are not the only multivariate regression methods that exist; one could choose from a whole host of basis functions, non-parametric predictors, etc. We chose these methods since we consider them to be a fair comparison of "simple" regression methods. We implemented these methods using the `fitlm` function in Matlab with corresponding model specification: `interactions`, `purequadratic`, and `quadratic`, respectively. To solve *linear order statistic* (LOS) regression [30], we first sort the inputs and then apply a linear regression. The sorting process induces the non-linearity of LOS. The CIR β model that is used is indicated by CIR(b) where b indicates the number of bias terms $\{1, d, 2^d\}$—see Table II.

TABLE III: Regression Methods

Name	Model	Model Dim.	Comments
Linear	$y = \mathbf{w}^T\mathbf{h} + \beta$	$d+1$	Standard method
Interact	$y = \mathbf{w}^T\phi(\mathbf{h}) + \beta$	d^2+1	Linear plus pair-products (no square terms)
PureQuad	$y = \mathbf{w}^T\phi(\mathbf{h}) + \beta$	$2d+1$	Linear plus square terms (no *Interact* terms)
Quad	$y = \mathbf{w}^T\phi(\mathbf{h}) + \beta$	d^2+2d+1	Linear plus *Interact* and *PureQuad* terms
LOS	$y = \mathbf{w}^T\mathbf{h}_\pi + \beta$	$d+1$	Sorts input first
CIR(b)	$y = C_f(\mathbf{h})$	$2^d + b$	b indicates bias model $\{1, d, 2^d\}$

A. Impact of ridge regression on CIR methods

Using 10 real-world data sets from the UCI machine learning repository [29], we compared the performance of CIR methods with and without the application of ridge regression. Table IV shows the performance of the CIR(b) methods[8] and their corresponding ridge-regularized methods. The *mean squared error* (MSE) values presented in the table are the average values taken over 100 experimental trails—MSE of each trial is taken over a 10-fold cross validation. Based on a two-sample t-test, we identified the instances where the ridge regularization has improved the performance at a 5% statistical significance level. Overall, the application of ridge regression has improved the performance of CIR(1) in six out of ten instances, and for both the CIR(d) and CIR(2^d) methods, we observed improved performance in eight out of ten instances.

We increased the ridge regularization parameter (λ) in small steps and observed the MSE as well as the shrinkage of the learned regression parameters (i.e., the regression models sifted from the BC via (17)) for the CIR methods. The shrinkage plots in Figs. 1 and 2 demonstrate how the regularization

[7] The ridge regression minimizer, when learning regression weight vectors directly is $\hat{\mathbf{w}}_R = \left(X^T X + \lambda I\right)^{-1} X^T\mathbf{y}$, where X is a data (or design) matrix, $\mathbf{y}$ is a vector of target outputs, and I is the identity matrix [1].

[8] Since the number of parameters in the BC for CIR methods scales as 2^d where d is the number of features, computation becomes intractable for larger values of d, e.g., 10!=3,628,800. Therefore, for the data sets with more than six features, we applied *principal component analysis* (PCA) to reduce the dimensionality to six features, and then applied the CIR methods.

TABLE IV: Impact of ridge regression on CIR methods*

Method	Concrete	Real Estate	Fish Toxicity	Aquatic Toxicity	Red Wine	White Wine	ENB-2	Yacht	Airfoil	ISE
n	1,030	414	908	546	1599	4898	768	308	1503	536
d	8	5	6	8	11	11	8	6	5	7
CIR(1)	0.315 (0.003)	0.364 (0.008)	0.421 (0.007)	0.629 (0.333)	0.658 (0.004)	0.729 (0.001)	0.056 (0.001)	0.157 (0.041)	0.338 (0.002)	0.534 (0.012)
CIR(1)-Ridge	**0.311 (0.003)** λ = 0.01	**0.345 (0.004)** λ = 0.1	**0.410 (0.004)** λ = 0.1	0.513 (0.004) λ = 0.1	**0.648 (0.004)** λ = 0.1	**0.726 (0.001)** λ = 0.1	0.055 (0.001) λ = 0.001	0.150 (0.006) λ = 0.0001	0.338 (0.002) λ = 0.001	**0.479 (0.006)** λ = 0.1
CIR(d)	0.311 (0.004)	0.365 (0.007)	0.423 (0.015)	0.618 (0.122)	0.655 (0.004)	0.725 (0.002)	0.052 (0.001)	0.151 (0.015)	0.336 (0.001)	0.542 (0.015)
CIR(d)-Ridge	**0.307 (0.003)** λ = 0.01	**0.347 (0.004)** λ = 0.1	**0.410 (0.003)** λ = 0.1	**0.515 (0.005)** λ = 0.1	**0.645 (0.003)** λ = 0.1	**0.723 (0.001)** λ = 0.1	**0.055 (0.001)** λ = 0.0001	0.145 (0.008) λ = 0.0001	0.336 (0.002) λ = 0.0001	**0.487 (0.005)** λ = 0.1
CIR(2^d)	0.302 (0.005)	0.358 (0.011)	0.462 (0.011)	0.800 (0.101)	0.683 (0.007)	0.724 (0.002)	0.222 (0.412)	0.189 (0.031)	0.315 (0.002)	0.630 (0.021)
CIR(2^d)-Ridge	**0.294 (0.004)** λ = 0.01	**0.341 (0.007)** λ = 0.1	**0.436 (0.005)** λ = 0.1	**0.546 (0.010)** λ = 0.1	**0.670 (0.005)** λ = 0.1	**0.721 (0.002)** λ = 0.1	0.053 (0.001) λ = 0.001	**0.152 (0.018)** λ = 0.001	0.315 (0.002) λ = 0.001	**0.552 (0.010)** λ = 0.1

*MSE values in the table are the average of 100 experimental trails—MSE of each trial is taken over a 10-fold cross validation. Bold indicates that the ℓ_2-regularization has improved the performance of that CIR method at a 5% statistical significance.

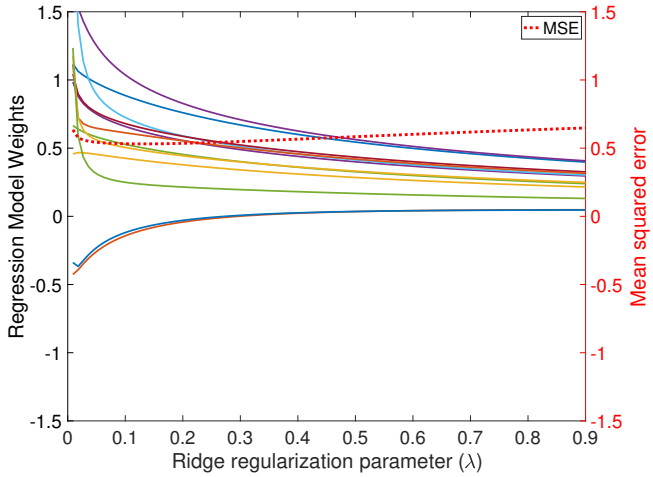

Fig. 1: Impact of ridge regularization on the CIR(1) BC parameters learned on Aquatic Toxicity data set.

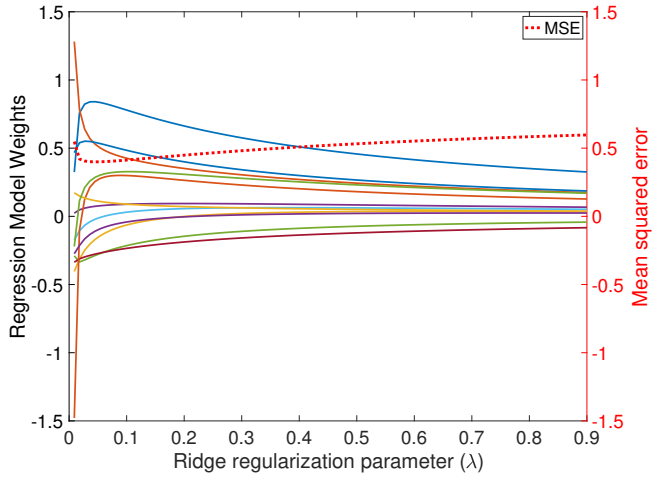

Fig. 2: Impact of ridge regularization on the CIR(1) BC parameters learned on Istanbul Stock Exchange data set.

restricted the magnitude of the parameters[9] with the increasing λ values. In both the examples, the best MSE was observed between the λ values of 0.01 and 0.1.

B. Performance of CIR methods

Table V shows the regression results of the CIR methods and the comparison algorithms for several real-world data sets. For each data set, the table shows the number of objects n and the number of features d. The best algorithms for each data set are shown in bold font. We performed a two-sample t-test at a 5% significance level to determine the statistically best results; hence, more than one algorithm can be considered as best. Overall, CIR methods produced best results on six out of 10 data sets, and CIR(1) with ridge regression was the best algorithm with best results on four out of 10 data sets. Even for the cases where the CIR methods were not the best, the ridge regularized CIR methods produced good(-enough) results.

[9]The data sets Aquatic Toxicity and Instanbul Stock Exchange have eight features and seven features respectively. However, since we applied PCA to reduce the dimensionality to six features, the learned BCs for both the data sets have 2^6=64 parameters. In Figs. 1 and 2, for demonstration purpose, we are showing the trend of a randomly selected 12 out of the 64 BC parameters.

VI. Conclusions and Future Work

In this work, we reviewed our previously developed method of learning regression models using the Choquet fuzzy integral [2] and enhanced this method by applying ℓ_2 regularization. Experimental results with real-world data sets demonstrated that the introduction of regularization significantly improved the performance of CIR methods in 22 out of 30 (73%)

TABLE V: MSE on Benchmark Data Sets*

Method	Concrete	Real Estate	Fish Toxicity	Aquatic Toxicity	Red Wine	White Wine	ENB-2	Yacht	Airfoil	ISE	# of best instances
n	1,030	414	908	546	1599	4898	768	308	1503	536	-
d	8	5	6	8	11	11	8	6	5	7	-
Linear	0.424 (0.002)	0.445 (0.002)	0.437 (0.002)	0.553 (0.003)	0.661 (0.001)	0.729 (0.001)	0.087 (0.000)	0.520 (0.003)	0.505 (0.001)	**0.437 (0.003)**	1
Linear-ℓ_1	0.418 (0.003) λ = 0.01	0.445 (0.002) λ = 0.01	0.435 (0.002) λ = 0.01	0.550 (0.005) λ = 0.01	0.660 (0.001) λ = 0.01	0.728 (0.001) λ = 0.01	0.087 (0.001) λ = 0.0001	0.519 (0.004) λ = 0.0001	0.501 (0.001) λ = 0.01	**0.437 (0.003)** λ = 0.001	1
Linear-ℓ_2	0.411 (0.002) λ = 0.1	0.445 (0.002) λ = 0.01	0.434 (0.002) λ = 0.1	0.552 (0.004) λ = 0.1	0.658 (0.001) λ = 0.1	0.728 (0.001) λ = 0.1	0.087 (0.001) λ = 0.0001	0.519 (0.004) λ = 0.0001	0.493 (0.001) λ = 0.1	**0.437 (0.003)** λ = 0.001	1
Interactions	0.358 (0.003)	0.364 (0.004)	0.417 (0.004)	0.764 (0.105)	0.658 (0.004)	0.763 (0.004)	0.053 (0.000)	0.409 (0.009)	0.371 (0.001)	0.485 (0.009)	0
Quadratic	**0.229 (0.001)**	0.349 (0.005)	0.424 (0.005)	0.714 (0.141)	0.657 (0.005)	0.774 (0.007)	**0.011 (0.000)**	0.087 (0.002)	0.367 (0.002)	0.509 (0.010)	2
Pure Quadratic	0.264 (0.001)	0.367 (0.004)	0.428 (0.003)	0.552 (0.007)	0.665 (0.003)	0.758 (0.002)	0.078 (0.000)	**0.080 (0.001)**	0.454 (0.001)	0.462 (0.006)	1
LOS	0.708 (0.002)	0.727 (0.005)	0.718 (0.003)	0.940 (0.007)	0.996 (0.002)	0.977 (0.001)	0.758 (0.003)	1.086 (0.009)	0.797 (0.001)	0.508 (0.008)	0
LOS-ℓ_1	0.707 (0.002) λ = 0.01	0.727 (0.007) λ = 0.001	0.709 (0.002) λ = 0.01	0.934 (0.007) λ = 0.01	0.987 (0.011) λ = 0.0001	0.977 (0.001) λ = 0.0001	0.754 (0.003) λ = 0.01	1.088 (0.006) λ = 0.0001	0.797 (0.002) λ = 0.01	0.509 (0.006) λ = 0.1	0
LOS-ℓ_2	0.704 (0.002) λ = 0.01	0.723 (0.006) λ = 0.01	0.716 (0.003) λ = 0.01	0.935 (0.005) λ = 0.1	0.985 (0.012) λ = 0.0001	0.977 (0.001) λ = 0.0001	0.748 (0.004) λ = 0.01	1.086 (0.007) λ = 0.0001	0.790 (0.001) λ = 0.1	0.505 (0.004) λ = 0.01	0
CIR(1)	0.315 (0.003)	0.364 (0.008)	0.421 (0.007)	**0.629 (0.333)**	0.658 (0.004)	0.729 (0.001)	0.056 (0.001)	0.157 (0.041)	0.338 (0.002)	0.534 (0.012)	1
CIR(d)	0.311 (0.004)	0.365 (0.007)	0.423 (0.015)	0.618 (0.122)	0.655 (0.004)	0.725 (0.002)	0.052 (0.001)	0.151 (0.015)	0.336 (0.001)	0.542 (0.015)	0
CIR(2^d)	0.302 (0.005)	0.358 (0.011)	0.462 (0.011)	0.800 (0.101)	0.683 (0.007)	0.724 (0.002)	0.222 (0.412)	0.189 (0.031)	**0.315 (0.002)**	0.630 (0.021)	1
CIR(1)-Ridge	0.311 (0.003) λ = 0.01	**0.345 (0.004)** λ = 0.1	**0.410 (0.004)** λ = 0.1	**0.513 (0.004)** λ = 0.1	**0.648 (0.004)** λ = 0.1	0.726 (0.001) λ = 0.1	0.055 (0.001) λ = 0.001	0.150 (0.006) λ = 0.0001	0.338 (0.002) λ = 0.001	0.479 (0.006) λ = 0.1	4
CIR(d)-Ridge	0.307 (0.003) λ = 0.01	0.347 (0.004) λ = 0.1	**0.410 (0.003)** λ = 0.1	**0.515 (0.005)** λ = 0.1	**0.645 (0.003)** λ = 0.1	0.723 (0.001) λ = 0.1	0.055 (0.001) λ = 0.0001	0.145 (0.008) λ = 0.0001	0.336 (0.002) λ = 0.0001	0.487 (0.005) λ = 0.1	3
CIR(2^d)-Ridge	0.294 (0.004) λ = 0.01	**0.341 (0.007)** λ = 0.1	0.436 (0.005) λ = 0.1	0.546 (0.010) λ = 0.1	0.670 (0.005) λ = 0.1	**0.721 (0.002)** λ = 0.1	0.053 (0.001) λ = 0.001	0.152 (0.018) λ = 0.001	**0.315 (0.002)** λ = 0.001	0.552 (0.010) λ = 0.1	3

*MSE values in the table are the average of 100 experimental trails—MSE of each trial is taken over a 10-fold cross validation. Bold indicates lowest MSE at a 5% statistical significance.

instances. We then showed that ℓ_2 regularized CIR methods were superior to competing regression models on real-world benchmark data.

Future work will include extending the CIR method to deep learning architectures. In a recently published work [31] we show that the Choquet integral can be built using standard neural network operations and can thus be used to represent learned layers in deep networks. We also will extend the *explainable AI* (XAI) approaches proposed in [32, 33], which allow interpretation of the result of the Choquet integral.

ACKNOWLEDGEMENT

This work was supported in part by the MTU Institute of Computing and Cybersystems. Superior, a high-performance computing infrastructure at Michigan Technological University, was used in obtaining results presented in this publication.

REFERENCES

[1] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

[2] T. C. Havens and D. T. Anderson, "Machine learning of choquet integral regression with respect to a bounded capacity (or non-monotonic fuzzy measure)," in *IEEE Int. Conf. Fuzzy Systems*, 2019.

[3] M. Grabisch, T. Murofushi, and M. Sugeno, Eds., *Fuzzy Measures and Integrals: Theory and Applications*. New York: Physica-Verlag, 2000.

[4] M. Grabisch, "Modelling data by the choquet integral," in *Information Fusion in Data Mining*, ser. Studies in Fuzziness and Soft Compting, V. Torra, Ed. Berlin, Heidelberg: Springer, 2003, vol. 123.

[5] H.-C. Liu, W.-S. Chen, Y.-C. Tu, and Y.-K. Yu, "Choquet integral regression model based on high-order l-measure," in *Proc. Int. Conf. Mach. Learn. and Cyber.*, Jul. 2009, pp. 3177–3182.

[6] A. F. Tehrani, W. Cheng, and E. Hüllermeier, "Choquistic regression: Generalizing logistic regression using the choquet integral," in *Proc. EUSFLAT*, 2011, pp. 868–875.

[7] X. Du and A. Zare, "Multiple instance choquet integral classifier fusion and regression for remote sensing applications," *CoRR*, vol. abs/1803.04048, 2018.

[8] T. Murofushi, M. Sugeno, and M. Machida, "Non-monotonic fuzzy measures and the Choquet integral," *Fuzzy Sets and Systems*, vol. 64, pp. 73–86, 1994.

[9] A. D. Waegenaere and P. P. Wakker, "Nonmonotonic Choquet integrals," *J. Mathematical Economics*, vol. 36, pp. 45–60, 2001.

[10] Y. Narukawa, T. Murofushi, and M. Sugeno, "Space of fuzzy measures and convergence," *Fuzzy sets and systems*, vol. 138, no. 3, pp. 497–506, 2003.

[11] Y. Narukawa and V. Torra, "Non-monotonic fuzzy measures and intuitionistic fuzzy sets," in *Modeling Decisions for Artificial Intelligence*, ser. Lecture Notes in Computer Science, V. Torra, Y. Narukawa, A. Valls, and J. Domingo-Ferrer, Eds. Berlin, Heidelberg: Springer, 2006, vol. 3885.

[12] M. Grabisch, *Fuzzy Measures and Integrals: Theory and Applications*. New York: Physica-Verlag, 2000, ch. Fuzzy integral for classification and feature extraction, pp. 415–434.

[13] J. M. Keller, P. Gader, and A. K. Hocaoglu, *Fuzzy Measures and Integrals: Theory and Applications*. New York: Physica-Verlag, 2000, ch. Fuzzy integral in image processing and recognition, pp. 435–466.

[14] S. Auephanwiriyakul, J. M. Keller, and P. Gader, "Generalized Choquet fuzzy integral fusion," *Information Fusion*, vol. 3, pp. 69–85, 2002.

[15] H. Tahani and J. M. Keller, "Information fusion in computer vision using the fuzzy integral," *IEEE Trans. Systems Man Cybernet.*, vol. 20, no. 3, pp. 733–741, 1990.

[16] M. Anderson, D. T. Anderson, and D. Wescott, "Estimation of adult skeletal age-at-death using the Sugeno fuzzy integral," *American Journal of Physical Anthropology*, vol. 142, no. 1, pp. 30–41, 2009.

[17] D. T. Anderson, T. C. Havens, C. Wagner, J. M. Keller, M. F. Anderson, and D. J. Wescott, "Extension of the fuzzy integral for general fuzzy set-valued information," *IEEE Trans. Fuzzy Systems*, vol. 22, no. 6, pp. 1625–1639, 2014.

[18] G. Choquet, "Theory of capacities," *Analles de l'Institit Fourier*, vol. 5, pp. 131–295, 1953.

[19] T. Murofushi and M. Sugeno, "An interpretation of fuzzy measure and the Choquet integral as an integral with respect to a fuzzy measure," *Fuzzy Sets and Systems*, vol. 29, no. 2, pp. 201–227, 1989.

[20] M. Grabisch, H. T. Nguyen, and E. A. Walker, *Fundamentals of Uncertainty Calculi, With Applications to Fuzzy Inference*. Dordrecht: Kluwer Academic, 1995.

[21] M. Sugeno, "Theory of fuzzy integral and its applications," Ph.D. dissertation, Tokyo Institute of Technology, 1974.

[22] M. Grabisch, "Fuzzy integral in multicriteria decision making," *Fuzzy Sets and Systems*, vol. 69, pp. 279–298, 1995.

[23] D. T. Anderson, J. M. Keller, and T. C. Havens, "Learning fuzzy-valued fuzzy measures for the fuzzy-valued Sugeno fuzzy integral," in *Proc. IPMU*, Dortmund, Germany, 2010.

[24] D. T. Anderson, S. Price, and T. C. Havens, "Regularization-based learning of the choquet integral," in *Proc. IEEE Int. Conf. Fuzzy Systems*, 2014.

[25] A. N. Tikhonov, "On the stability of inverse problems," in *Dokl. Akad. Nauk SSSR*, vol. 39, 1943, pp. 195–198.

[26] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, 2006.

[27] L. Jacob, G. Obozinski, and J.-P. Vert, "Group lasso with overlap and graph lasso," in *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009, pp. 433–440.

[28] L. Yuan, J. Liu, and J. Ye, "Efficient methods for overlapping group lasso," in *Advances in Neural Information Processing Systems*, 2011, pp. 352–360.

[29] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[30] R. R. Yager and D. P. Filev, "Induced ordered weighted averaging operators," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 29, no. 2, pp. 141–150, 1999.

[31] M. A. Islam, D. T. Anderson, A. Pinar, T. C. Havens, G. Scott, and J. M. Keller, "Enabling explainable fusion in deep learning with fuzzy integral neural networks," *IEEE Transactions on Fuzzy Systems*, 2019.

[32] S. Price, D. T. Anderson, C. Wagner, T. C. Havens, and J. M. Keller, "Indices for introspection of the choquet integral," in *Studies in Fuzziness and Soft Computing*, ser. Proc. World Conf. Soft Computing, vol. 312, 2014, pp. 261–271.

[33] B. Murray, M. A. Islam, A. J. Pinar, D. T. Anderson, T. C. Havens, and G. Scott, "Explainable ai for understanding decisions and data-driven optimization of the choquet integral," in *Proc. IEEE Int. Conf. Fuzzy Sys.*, 2018.