melt: Multiple Empirical Likelihood Tests in R

Eunseop Kim
The Ohio State University

Steven N. MacEachern
The Ohio State University

Mario Peruggia
The Ohio State University

Abstract

Empirical likelihood enables a nonparametric, likelihood-driven style of inference without relying on assumptions frequently made in parametric models. Empirical likelihood-based tests are asymptotically pivotal and thus avoid explicit studentization. This paper presents the R package **melt** that provides a unified framework for data analysis with empirical likelihood methods. A collection of functions are available to perform multiple empirical likelihood tests for linear and generalized linear models in R. The package **melt** offers an easy-to-use interface and flexibility in specifying hypotheses and calibration methods, extending the framework to simultaneous inferences. Hypothesis testing uses a projected gradient algorithm to solve constrained empirical likelihood optimization problems. The core computational routines are implemented in C++, with OpenMP for parallel computation.

Keywords: empirical likelihood, generalized linear models, hypothesis testing, optimization, R.

1. Introduction

The likelihood is an essential component of statistical inference. In a nonparametric or semi-parametric setting, where the quantity of interest is finite-dimensional, the maximum likelihood approach is not applicable since the underlying data-generating distribution is left unspecified. A popular approach in this context is the method of moments or the two-step generalized method of moments (GMM) (Hansen 1982) where only partial information is specified by moment conditions. Various one-step alternatives to GMM have been proposed over the last decades in the statistics and econometrics literatures (see, e.g., Efron 1981; Imbens 1997; Newey and Smith 2004).

One such alternative is empirical likelihood (EL) (Owen 1988, 1990; Qin and Lawless 1994). EL defines a likelihood function by profiling a nonparametric likelihood subject to the moment restrictions. While it is nonparametric in nature, some desirable properties of parametric likelihood apply to EL. Most notably, the EL ratio functions have limiting chi-square distributions under certain conditions. Without explicit studentization, confidence regions for the parameters can be constructed in much the same way as with a parametric likelihood. As the name suggests, however, the empirical distribution of the data determines the shape of the confidence region. Also, coverage accuracy of the confidence region can further be improved in principle, since EL is Bartlett-correctable (DiCiccio, Hall, and Romano 1991). In terms of estimation, the standard expansion argument (e.g., Yuan and Jennrich 1998; Jacod and Sørensen 2018) establishes the consistency and asymptotic normality of the maximum

empirical likelihood estimator (MELE). Moreover, Newey and Smith (2004) showed that the MELE generally has a smaller bias than its competitors and achieves higher-order efficiency after bias correction. EL methods have been extended to other areas, including linear models (Owen 1991), generalized linear models (Kolaczyk 1994; Chen and Cui 2003), survival analysis (Li, Li, and Zhou 2005), time series models (Kitamura 1997; Nordman and Lahiri 2014), and high-dimensional data analysis (Chen, Peng, and Qin 2009; Hjort, McKeague, and van Keilegom 2009). For an overview of EL and its applications, see Owen (2001) and Chen and van Keilegom (2009).

In the R language (R Core Team 2022), some software packages implementing EL and related methods are available from the Comprehensive R Archive Network (CRAN). The **emplik** package (Zhou 2022) provides a wide range of functions for analyzing censored and truncated data with EL. Confidence intervals for a one-dimensional parameter can also be constructed. Other examples and applications of the package can be found in Zhou (2015). The **emplik2** package (Barton 2022) is an extension for the two sample problems. Both packages cover the methods for the mean with uncensored data, which is the simplest case in terms of computation. In addition, the **EL** package (Valeinis and Cers 2022) performs EL tests for the difference between two sample means and the difference between two smoothed Huber estimators. The **eel** package (Wu and Zhang 2015) implements the extended empirical likelihood method (Tsao and Wu 2013) that expands the domain of EL to the full parameter space by applying a similarity transformation. It escapes the so-called "convex hull constraint" of EL that confines the domain to a bounded region. In fact, the gradient of log EL ratio functions diverges at the boundary. Using this property, the **elhmc** package (Kien, Chaudhuri, and Wei 2017) contains a single function `ELHMC` for Hamiltonian Monte Carlo sampling in Bayesian EL computation (Chaudhuri, Mondal, and Yin 2017). The **ELCIC** package (Shen and Wang 2022) develops an EL-based consistent information criterion in a model selection framework. The methods are relevant to longitudinal data. In a broader context of GMM and generalized empirical likelihood (Smith 1997), a few packages can be used for estimation with EL. The **gmm** package (Chaussé 2010) provides flexibility in specifying moment conditions. Other than GMM and EL, continuous updating (Hansen, Heaton, and Yaron 1996) and several estimation methods that belong to the family of generalized empirical likelihood are available. The **gmm** package has been superseded by the **momentfit** package (Chaussé 2022), which adds exponential tilting (Kitamura and Stutzer 1997) estimation and methods for constructing two-dimensional confidence regions.

This paper presents the R package **melt** (Kim 2022) that performs multiple empirical likelihood tests for regression analysis. The primary focus of the package is on linear and generalized linear models, perhaps most commonly used with the `lm()` and `glm()` functions in R. The package considers only just-identified models where the number of moment conditions equals the number of parameters. Typical linear models specified by `formula` objects in R are just-identified. In this case, the MELE is identical to the maximum likelihood estimator, and the estimate is easily obtained using `lm.fit()` or `glm.fit()` in the **stats** package. The fitted model then serves as a basis for testing hypotheses, which is a core component of the package. EL-based tests do not involve standard errors and `vcov()` methods since they are asymptotically pivotal and thus avoid explicit studentization. For this reason it is challenging to incorporate EL methods directly into other packages that perform inferences for parametric models using `vcov()`. We aim to bridge the gap and provide an easy-to-use interface that enables applying the methods to tasks routinely accomplished in R. Standard

tests performed by `summary.lm()` and `summary.glm()` methods, such as significance tests of the coefficients and an overall F test or a chi-square test, are available. Furthermore, in line with `lht()` in the **car** package (Fox and Weisberg 2019) or `glht()` in the **multcomp** package (Hothorn, Bretz, and Westfall 2008), the user can specify linear hypotheses to be tested. Multiple testing procedures are provided to control the family-wise error rate. Constructing confidence intervals and detecting outliers in a fitted model can also be done, adding more options for data analysis. Note that all the tests and methods rely on EL and its asymptotic properties. Although conceptually advantageous over parametric methods, this can lead to poor finite sample performance. Therefore, several calibration techniques are implemented in **melt** to mitigate this drawback of EL.

The rest of the paper is organized as follows. Section 2 describes EL methods and computational aspects of testing hypotheses with EL. Section 3 provides an overview of the **melt** package. Section 4 shows the basic usage of **melt** with implementation details. Examples are included to illustrate the applications of the package. We conclude with a summary and directions for future development in Section 5.

2. Background

2.1. Empirical likelihood framework

We describe a general framework for EL formulation. Suppose we observe independent and identically distributed (i.i.d.) p-dimensional random variables $X_1, \ldots, X_n$ from a distribution P. Consider a parameter $\theta \equiv \theta(P) \in \Theta$ and an estimating function $g(X_i, \theta)$ that take their values in $\mathbb{R}^p$ and satisfy the following moment condition:

$$\mathsf{E}[g(X_i, \theta)] = 0, \tag{1}$$

where the expectation is taken with respect to P. Without further information on P, we restrict our attention to a family of multinomial distributions supported on the data. The nonparametric likelihood is given by

$$L(P) = \prod_{i=1}^{n} P(\{X_i\}) = \prod_{i=1}^{n} p_i,$$

and the empirical distribution P_n maximizes L with $L(P_n) = n^{-n}$. Then the (profile) EL ratio function is defined as

$$R(\theta) = \max_{p_i} \left\{ \prod_{i=1}^{n} np_i : \sum_{i=1}^{n} p_i g(X_i, \theta) = 0,\ p_i \geq 0,\ \sum_{i=1}^{n} p_i = 1 \right\}, \tag{2}$$

with $L(\theta) = \prod_{i=1}^{n} p_i$ denoting the corresponding EL function. The profiling removes all the nuisance parameters, the p_is attached to the data, yielding a p-dimensional subfamily indexed by θ. Note that the data determine the multinomial distributions; thus, the reduction to a subfamily does not correspond to a parametric model. See DiCiccio and Romano (1990) for a detailed discussion of how the reduction connects to the notion of least favorable families (Stein 1956).

We maximize $\prod_{i=1}^n np_i$, or equivalently $\sum_{i=1}^n \log(np_i)$, subject to the constraints in Equation 2. The convex hull constraint refers to the condition that the convex hull of the points $\{g(X_i,\theta)\}_{i=1}^n$ contains the zero vector. If the constraint is not satisfied, the problem is infeasible as some p_is are forced to be negative to match the condition that $\sum_{i=1}^n p_i g(X_i,\theta) = 0$. Otherwise, the problem admits a unique interior solution since the objective function is concave and the feasible set is convex. Using the method of Lagrange multipliers, we write

$$\mathcal{L}(p_1,\ldots,p_n,\lambda,\gamma) = \sum_{i=1}^n \log(np_i) - n\lambda^\top \sum_{i=1}^n p_i g(X_i,\theta) + \gamma\left(\sum_{i=1}^n p_i - 1\right),$$

where λ and γ are the multipliers. Differentiating $\mathcal{L}$ with respect to its arguments and setting the derivatives to zero gives $\gamma = -n$. Then the solution is given by

$$p_i = \frac{1}{n}\frac{1}{1+\lambda^\top g(X_i,\theta)}, \tag{3}$$

where $\lambda \equiv \lambda(\theta)$ satisfies

$$\frac{1}{n}\sum_{i=1}^n \frac{g(X_i,\theta)}{1+\lambda^\top g(X_i,\theta)} = 0. \tag{4}$$

Instead of solving the nonlinear Equation 4, we solve the dual problem with respect to λ. Substituting the expression for p_i in Equation 3 into Equation 2 gives

$$\log\left(R(\theta)\right) = -\sum_{i=1}^n \log\left(1+\lambda^\top g(X_i,\theta)\right) =: r(\lambda). \tag{5}$$

Now consider minimizing $r(\lambda)$ subject to $1+\lambda^\top g(X_i,\theta) \geq 1/n$ for $i = 1,\ldots,n$ with θ fixed. This is a convex optimization problem, where the constraints correspond to the condition that $0 \leq p_i \leq 1$ for all i. Next, we remove the constraints by employing a pseudo logarithm function (Owen 2001)

$$\log_\star(x) = \begin{cases} \log(x), & \text{if } x \geq 1/n, \\ -n^2x^2/2 + 2nx - \log(n) - 3/2, & \text{if } x < 1/n. \end{cases} \tag{6}$$

Minimizing $r_\star(\lambda) = -\sum_{i=1}^n \log_\star(1+\lambda^\top g(X_i,\theta))$ without the constraints does not affect the solution and the Newton-Raphson method can be applied to find it. If the convex hull constraint is violated, the algorithm does not converge with $\|\lambda\|$ increasing as the iteration proceeds. Hence, it can be computationally more efficient to minimize $r_\star(\lambda)$ first to get $\log(R(\theta))$ and indirectly check the convex hull constraint by observing λ and p_is. Note that EL is maximized when $\lambda = 0$ and $p_i = 1/n$ for all i. It follows from Equation 4 that $\hat{\theta}$, the MELE, is obtained by solving the estimating equations $\sum_{i=1}^n g(X_i,\theta) = 0$. The existence, uniqueness, and asymptotic properties of $\hat{\theta}$ are well established in the literature.

Assume that there exists $\theta_0 \in \Theta$ that is the unique solution to the moment condition in Equation 1. Similar to the parametric likelihood method, define minus twice the empirical log-likelihood ratio function as $l(\theta) = -2\log(R(\theta))$. Under regularity conditions, it is known that the a nonparametric version of Wilks' theorem holds. That is, $l(\theta_0) \to_d \chi_p^2$ as $n \to \infty$, where χ_p^2 is the chi-square distribution with p degrees of freedom. See, e.g., Qin and Lawless (1994) for proof and the treatment of more general cases, including the over-identified ones. For a

level $\alpha \in (0,1)$, let $\chi^2_{p,\alpha}$ be the $100(1-\alpha)$%th percentile of χ^2_p. Since $P(l(\theta_0) \leq \chi^2_{p,\alpha}) \to 1-\alpha$, an asymptotic $100(1-\alpha)$% confidence region for θ can be obtained as

$$\left\{\theta : l(\theta) \leq \chi^2_{p,\alpha}\right\}. \tag{7}$$

Often the chi-square calibration is unsatisfactory due to slow convergence, especially when n is small. We review other calibration methods that address this issue. First, consider EL for the mean with $g(X_i,\theta) = X_i - \theta$ and $\theta_0 = \mathsf{E}[X_i]$. Then we have $l(\theta_0) = n(\bar{X}-\theta_0)^\top V^{-1}(\bar{X}-\theta_0) + o_P(1)$, where $V = n^{-1}\sum_{i=1}^n (X_i-\theta_0)(X_i-\theta_0)^\top$. Let $S = (n-1)^{-1}\sum_{i=1}^n (X_i-\bar{X})(X_i-\bar{X})^\top$ and define a Hotelling's T squared statistic as $T^2 = n(\bar{X}-\theta_0)^\top S^{-1}(\bar{X}-\theta_0)$. It can be shown that $n(\bar{X}-\theta_0)^\top V^{-1}(\bar{X}-\theta_0) = nT^2/(T^2+n-1) = T^2 + O_P(n^{-1})$. This suggests that we can use $p(n-1)F_{p,n-p,\alpha}/(n-p)$ in place of $\chi^2_{p,\alpha}$, where $F_{p,n-p}$ is a F distribution with p and $n-p$ degrees of freedom. The F calibration yields a larger critical value than the chi-square calibration, which leads to a better coverage probability of the confidence region in Equation 7. Next, a more generally applicable calibration method is the Bartlett correction. Based on the Edgeworth expansion, it requires Cramér's condition and the existence of finite higher moments of $g(X_i,\theta)$. The correction is given by a scale multiple of $\chi^2_{p,\alpha}$ as $(1+a/n)\chi^2_{p,\alpha}$ with an unknown constant a. In practice, the Bartlett correction involves getting a consistent estimate $\hat{a}$ with plug-in sample moments. The coverage error of the Bartlett corrected confidence region is reduced from $O(n^{-1})$ to $O(n^{-2})$ (DiCiccio *et al.* 1991), which is unattainable by the F calibration. Another effective calibration method is the bootstrap. Let $\widetilde{\mathcal{X}}_n = \{\widetilde{X}_1,\ldots,\widetilde{X}_n\}$ denote the null-transformed data such that $\mathsf{E}_{P_n}[g(\widetilde{X}_i,\theta)] = n^{-1}\sum_{i=1}^n g(\widetilde{X}_i,\theta) = 0$, so Equation 1 holds for $\widetilde{\mathcal{X}}_n$ with P_n. Define a bootstrap sample $\widetilde{\mathcal{X}}^*_n = \{\widetilde{X}^*_1,\ldots,\widetilde{X}^*_n\}$ as i.i.d. observations from $\widetilde{\mathcal{X}}_n$. We can compute the bootstrap EL ratio $l^*(\theta)$ with $\widetilde{\mathcal{X}}^*_n$ in the same way as before. The critical value, the $100(1-\alpha)$%th percentile of the distribution of $l^*(\theta)$, can be approximated using a large number, say B, of bootstrap samples. As an example, we may set $\widetilde{X}_i = X_i - \bar{X} + \theta$ when $g(X_i,\theta) = X_i - \theta$. This is equivalent to computing $l^*(\bar{X})$ with a bootstrap sample from the observed data directly. The $O(n^{-2})$ coverage error can also be achieved by the bootstrap calibration (Hall and Scala 1990).

Although EL does not require full model specification, it is not entirely free of misspecification concerns. Developing diagnostic measures for EL is still an open problem, and we briefly introduce the technique of empirical likelihood displacement (ELD) (Lazar 2005). Much like the concept of likelihood displacement (Cook 1986), ELD can be used to detect influential observations or outliers. With the MELE $\hat{\theta}$ from the complete data, consider reduced data with the ith observation deleted and the corresponding MELE estimate $\hat{\theta}_{(i)}$, Then ELD is defined as

$$\text{ELD}_i = 2\left(L(\hat{\theta}) - L(\hat{\theta}_{(i)})\right), \tag{8}$$

where $\hat{\theta}_{(i)}$ is plugged into the original EL function $L(\theta)$. If ELD_i is large, the ith observation is an influential point and can be inspected as a possible outlier. See Zhu, Ibrahim, Tang, and Zhang (2008) for other diagnostic measures for EL.

2.2. Empirical likelihood for linear models

We now turn our attention to linear models, which are the main focus of **melt**. First, suppose we have independent observations $\{(Y_i,X_i)\}_{i=1}^n$, where Y_i is the univariate response and X_i is the p-dimensional vector of covariates (including the intercept, if any). For illustrative purposes, we consider X_i fixed and do not explicitly distinguish between random and

fixed designs. See Kitamura, Tripathi, and Ahn (2004) for formal methods for models with conditional moment restrictions. For standard linear regression models, assume that

$$\mathsf{E}[Y_i] = \mu_i,\ \mathsf{VAR}[Y_i] = \sigma_i^2,\ i = 1, \ldots, n,$$

where $\mu_i = X_i^\top \theta^*$ for some $\theta^* \in \mathbb{R}^p$. Since θ^* minimizes $\mathsf{E}[(Y_i - X_i^\top \theta)^2]$, we have the following moment conditions

$$\mathsf{E}[(Y_i - X_i^\top \theta)X_i] = 0,\ i = 1, \ldots, n,$$

and the estimating equations

$$\sum_{i=1}^{n} (Y_i - X_i^\top \theta)X_i = 0.$$

Let $Z_i = (Y_i, X_i)$ and $g(Z_i, \theta) = (Y_i - X_i^\top \theta)X_i$. The $g(Z_i, \theta)$s are independent with possibly nonconstant variances, regardless of whether the σ_i^2s are constant. Following the steps in Section 2.1, we can compute the EL ratio function

$$R(\theta) = \max_{p_i} \left\{ \prod_{i=1}^{n} np_i : \sum_{i=1}^{n} p_i g(Z_i, \theta) = 0,\ p_i \geq 0,\ \sum_{i=1}^{n} p_i = 1 \right\}. \tag{9}$$

Under mild moment conditions it follows that $l(\theta^*) \to_d \chi_p^2$. Note also from Equation 9 that the least square estimator $\hat{\theta}$ is the MELE for θ, with $L(\hat{\theta}) = n^{-n}$ and $R(\hat{\theta}) = 0$.

Next, generalized linear models assume that

$$\mathsf{E}[Y_i] = \mu_i,\ G(\mu_i) = X_i^\top \theta,\ \mathsf{VAR}[Y_i] = \phi V(\mu_i),\ i = 1, \ldots, n,$$

where G and V are known link and variance functions, respectively, and ϕ is an optional dispersion parameter. EL for generalized linear models builds upon quasi-likelihood methods (Wedderburn 1974). The log quasi-likelihood for Y_i is given by

$$Q(Y_i, \mu_i) = \int_{Y_i}^{\mu_i} \frac{Y_i - t}{\phi V(t)} dt.$$

Differentiating $Q(Y_i, \mu_i)$ with respect to θ yields the quasi-score

$$\frac{H'(X_i^\top \theta)\left(Y_i - H(X_i^\top \theta)\right)}{\phi V\left(H(X_i^\top \theta)\right)} X_i =: g_1(Z_i, \theta),$$

where H denotes the inverse link function. From $\mathsf{E}[g_1(Z_i, \theta^*)] = 0$ for $i = 1, \ldots, n$, we get the estimating equations

$$\sum_{i=1}^{n} g_1(Z_i, \theta) = 0.$$

Then the EL ratio function can be derived as in Equation 9 with the same asymptotic properties. It can be seen that the MELE for θ is the same as the quasi-maximum likelihood estimator. When overdispersion is present with unknown ϕ, we introduce another estimating function based on the squared residuals. Let $\eta = (\theta, \phi)$ and

$$g_2(Z_i, \eta) = \frac{\left(Y_i - H(X_i^\top \theta)\right)^2}{\phi^2 V\left(H(X_i^\top \theta)\right)} - \frac{1}{\phi}, \tag{10}$$

where $\mathsf{E}[g_2(Z_i, \eta^*)] = 0$ for some $\eta^* = (\theta^*, \phi^*)$. We compute the EL ratio function with this additional constraint as

$$R(\eta) = \max_{p_i} \left\{ \prod_{i=1}^{n} np_i : \sum_{i=1}^{n} p_i g_1(Z_i, \eta) = 0,\ \sum_{i=1}^{n} p_i g_2(Z_i, \eta) = 0,\ p_i \geq 0,\ \sum_{i=1}^{n} p_i = 1 \right\}.$$

The computation is straightforward since the number of parameters equals the number of constraints on the estimating functions. Confidence regions for θ can be constructed by applying a calibration method to $l(\theta)$. One advantage of using EL for linear models is that the confidence regions have data-driven shapes and orientations.

2.3. Hypothesis testing with empirical likelihood

As seen in Section 2.2, it is easy to compute the MELE and evaluate the EL ratio function at a given value for linear models. Conducting significance tests, or hypothesis testing in general, is often the main interest when using a linear model. The EL method can be naturally extended to testing hypotheses by imposing appropriate constraints on the parameter space Θ (Qin and Lawless 1995; Adimari and Guolo 2010). Consider a null hypothesis $\mathcal{H}$ corresponding to a nonempty subset of Θ through a smooth q-dimensional function h such that $\mathcal{H} = \{\theta \in \Theta : h(\theta) = 0\}$. With additional conditions on $\mathcal{H}$ and h, it can be shown that

$$\inf_{\theta : h(\theta) = 0} l(\theta) \to_d \chi^2_q \tag{11}$$

under the null that $\theta^* \in \mathcal{H}$. In practice, computing the solution in Equation 11 is a nontrivial task. Recall that the convex hull constraint restricts the domain of $l(\theta)$ to $\Theta_n := \{\theta \in \Theta : 0 \in \mathrm{Conv}_n(\theta)\}$, where $\mathrm{Conv}_n(\theta)$ denotes the convex hull of $\{g(Z_i, \theta)\}_{i=1}^n$ with an estimating function g. Except for a few cases, both $l(\theta)$ and Θ_n are nonconvex in θ, and fully identifying Θ_n can be even more challenging than the constrained minimization problem itself. Given that the solution can only be obtained numerically by an iterative process, it is essential to monitor the entire solution path in $\mathrm{Conv}_n(\theta) \cap \mathcal{H}$. Another difficulty is in the nested optimization structure. The Lagrange multiplier λ needs to be updated for each update of θ, which amounts to solving an inner layer of optimization in Equation 5 at every step. It is clear that no single method can be applied to all estimating functions and hypotheses. Tang and Wu (2014) proposed a nested coordinate descent algorithm for general constrained EL problems, where the outer layer is optimized with respect to θ with λ fixed. After some algebra, we obtain for $\theta \in \Theta_n$ the gradient of the EL ratio function

$$\nabla \log\left(R(\theta)\right) = -\frac{1}{n} \sum_{i=1}^{n} \frac{1}{1 + \lambda^\top g(Z_i, \theta)} \partial_\theta g(Z_i, \theta) \lambda, \tag{12}$$

where $\partial_\theta g(Z_i, \theta)$ represents the Jacobian matrix of $g(Z_i, \theta)$. Observe that the expression does not involve any derivatives with respect to λ. In order to reduce the computational complexity, we focus only on linear hypotheses of the form

$$\mathcal{H} = \{\theta \in \Theta : L\theta = r\}, \tag{13}$$

where L is a $q \times p$ matrix and r is a q-dimensional vector. We use the projected gradient descent approach to obtain a local minimum of $l(\theta)$ in Equation 11. The projected gradient of $l(\theta)$ can

Algorithm 1: Constrained empirical likelihood optimization using the projected gradient descent.

Data: $X_1, \ldots, X_n$.

Input : $\theta_0 \in \text{Conv}_n(\theta) \cap \mathcal{H}$, $\lambda_0 \coloneqq \lambda(\theta_0)$, m (outer layer maximum number of iterations), ϵ (outer layer convergence tolerance), m_l (inner layer maximum number of iterations), ϵ_l (inner layer convergence tolerance), and P.

Output: Optimal θ that minimizes $l(\theta)$ subject to the constraint $\mathcal{H}$ and the corresponding λ.

$\theta \leftarrow \theta_0$; `// Assume that the initial value satisfies the constraints`
$\lambda \leftarrow \lambda_0$;
for $i \leftarrow 1$ **to** m **do** `// Outer layer`
  $\theta_{\text{temp}} \leftarrow \theta$;
  $\lambda_{\text{temp}} \leftarrow \lambda$;
  $\gamma \leftarrow 2$;
  while $l(\theta_{\text{temp}}) >= l(\theta)$ **do**
    $\gamma \leftarrow \gamma/2$;
    $\Delta \leftarrow -\gamma P \nabla l(\theta)$;
    $\theta_{\text{temp}} \leftarrow \theta + \gamma\Delta$;
    $\lambda_{\text{temp}} \leftarrow 0$;
    for $j \leftarrow 1$ **to** m_l **do** `// Inner layer`
      $\Delta_l \leftarrow -\left(\nabla^2 r_\star(\lambda_{\text{temp}})\right)^{-1} \nabla r_\star(\lambda_{\text{temp}})$;
      $\gamma_l \leftarrow 1$;
      while $r_\star(\lambda_{\text{temp}} + \gamma_l \Delta_l) > r_\star(\lambda_{\text{temp}})$ **do**
        $\gamma_l \leftarrow \gamma_l/2$;
      $\delta_l \leftarrow \|\lambda_{\text{temp}}\|$;
      $\lambda_{\text{temp}} \leftarrow \lambda_{\text{temp}} + \gamma_l \Delta_l$;
      if $\|\gamma_l \Delta_l\| < \epsilon_l \delta_l + \epsilon_l^2$ **then**
        break;
      else
        $j \leftarrow j + 1$;
  $\delta \leftarrow \|\theta_{\text{temp}}\|$;
  $\theta \leftarrow \theta_{\text{temp}}$;
  $\lambda \leftarrow \lambda_{\text{temp}}$;
  if $\|P\nabla l(\theta)\| < \epsilon$ **or** $\|\gamma\Delta\| < \epsilon\delta + \epsilon^2$ **then**
    break;
  else
    $i \leftarrow i + 1$;
return θ and λ;

be computed from Equation 12 with the orthogonal projector matrix $P = I_p - L^\top (LL^\top)^{-1} L$, where I_p denotes the $p \times p$ identity matrix. Then it would take a relatively small number of iterations for convergence, reducing the required number of inner layer updates of λ. The pseudo code is shown in Algorithm 1.

Controlling the type 1 error rate is necessary when testing multiple hypotheses simultaneously. Recently there has been interest in multiplicity-adjusted test procedures for Wald-type test statistics that asymptotically have a multivariate chi-square distribution under the global null hypothesis (Dickhaus and Royen 2015; Dickhaus and Sirotko-Sibirskaya 2019). Kim, MacEachern, and Peruggia (2021) proposed single-step multiple testing procedures for EL that asymptotically control the family-wise error rate with Monte Carlo simulations or bootstrap. Wang and Yang (2018) applied the F-calibrated EL statistics to the Benjamini-Hochberg procedure (Benjamini and Hochberg 1995) to control the false discovery rate.

3. Overview of melt

The latest stable release of **melt** is available from the CRAN at `https://CRAN.R-project.org/package=melt`. The development version, hosted by the rOpenSci, is on GitHub at `https://github.com/ropensci/melt`. Computational tasks are implemented in parallel using OpenMP (Dagum and Menon 1998) API in C++ with the **Rcpp** (Eddelbuettel and Balamuta 2018) and **RcppEigen** (Bates and Eddelbuettel 2013) packages to interface with R. Depending on the platform, the package can be compiled from source with support for OpenMP. The overall design of **melt** adopts the functional object-oriented programming approach (Chambers 2014) with `S4` classes and methods. Every function in the package is either a wrapper that creates a single instance of an object or a method that can be applied to a class object. The workflow of the package consists of three steps: (1) fitting a model, (2) examining and diagnosing the fitted model, and (3) testing hypotheses with the model. Four functions are available to build a model object whose names start with the prefix `el_`, which stands for empirical likelihood. A summary of the functions is provided below.

- `el_mean()`: creates an '`EL`' object for the mean.
- `el_sd()`: creates a '`SD`' object for the standard deviation.
- `el_lm()`: creates an '`LM`' object for the linear model.
- `el_glm()`: creates a '`GLM`' object for the generalized linear model. `el_glm()` does not support grouped data.

For univariate data, `el_mean()` corresponds to `t.test()` in the **stats** package. `el_lm()` and `el_glm()` correspond to `lm()` and `glm()` as well.

All model objects inherit from class '`EL`', and a description of the slots in '`EL`' is given in Table 1. Notably, the slot `optim` is a '`list`' with the following four components that summarize the optimization results:

- `par`: a numeric vector for the user-supplied parameter value θ where EL is evaluated.
- `lambda`: a numeric vector for the Lagrange multiplier λ.
- `iterations`: a single integer for the number of iterations performed.
- `convergence`: a single logical for the convergence status. It is either `TRUE` or `FALSE`.

Slot	Class	Description	Accessor
`optim`	`list`	Optimization results.	`getOptim()`
`logp`	`numeric`	Log probabilities of empirical likelihood.	`logProb()`
`logl`	`numeric(1)`	Empirical log-likelihood.	`logL()`
`loglr`	`numeric(1)`	Empirical log-likelihood ratio.	`logLR()`
`statistic`	`numeric(1)`	Minus twice the empirical log-likelihood ratio.	`chisq()`
`df`	`integer(1)`	Degrees of freedom associated with the `statistic`.	`getDF()`
`pval`	`numeric(1)`	p value of the `statistic`.	`pVal()`
`nobs`	`integer(1)`	Number of observations.	`nobs()`
`weights`	`numeric`	Re-scaled weights used for model fitting.	`weights()`
`coefficients`	`numeric`	MELE of the parameters.	`coef()`

Table 1: A description of some of the slots in an '`EL`' object. `numeric(1)` and `integer(1)` refer to a single numeric and integer, respectively. A full explanation of the class and slots can be found in the documentation of `EL-class` in the package.

Note that `par` is fixed in evaluating EL. The optimization is performed with respect to `lambda`, so `iterations` and `convergence` need to be understood in terms of `lambda`. Here we make a distinction between EL evaluation and EL optimization. The EL optimization refers to the constrained EL problem discussed in Section 2.3 and corresponds to another class '`CEL`' that directly extends '`EL`'. The `optim` slot in a '`CEL`' object has the same components. However, the optimization results are now interpreted in terms of `par`, the solution to the constrained problem. The '`LM`' and '`GLM`' classes contain '`CEL`', meaning that a constrained optimization is performed initially when `el_lm()` or `el_glm()` is called. In order to avoid confusion, the '`CEL`' class only distinguishes EL optimization from EL evaluation, and the user does not directly interact with a '`CEL`' object. Once `par` is obtained through evaluation or optimization, it uniquely determines `lambda` and, in turn, `logl` and `loglr`. Then `statistic` is equivalent to `-2 * loglr` and has an asymptotic chi-square distribution under the null hypothesis, with the associated `df` and `pval`. All four model fitting functions above accept an optional argument `weights` for weighted data. A vector of weights is then re-scaled internally for numerical stability in the computation of weighted EL (Glenn and Zhao 2007). Although `weights()` and `coef()` can extract `weights` and `coefficients`, these slots are mainly stored for subsequent analyses and methods.

In the next step, the following methods can be applied to an '`EL`' object to evaluate the model fit or compute summary statistics:

- `conv()`: extracts convergence status from a model. The distinction between the EL evaluation and EL optimization applies here as well. It can be used to check the convex hull constraint indirectly.
- `confint()`: computes confidence intervals for model parameters.
- `confreg()`: computes a two-dimensional confidence region for model parameters. It returns an object of class '`ConfregEL`' where a subsequent `plot()` method is applicable.

- `eld()`: computes empirical likelihood displacement in Equation 8 for model diagnostics and outlier detection. It returns an object of class '`ELD`' where a subsequent `plot()` method is applicable.

- `summary()`: summarizes the results of the overall model test and the significance tests for coefficients. Similar to `summary.lm()` and `summary.glm()`, it applies to a '`LM`' or '`GLM`' object and returns an '`SummaryLM`' or '`SummaryGLM`' object.

Lastly, we introduce the two main functions of **melt** that perform hypothesis testing. These generic methods take an '`EL`' object with other arguments that specify the problem in Equation 11.

- `elt()`: tests a linear hypothesis with EL. It returns an object of class '`ELT`' that contains the test statistic, the critical value, and the level of the test. Several calibration options discussed in Section 2.2 are available, and the p value is computed by the calibration method chosen.

- `elmt()`: tests multiple linear hypotheses simultaneously with EL. Each test can be considered as one instance of `elt()`. It returns an object of class '`ELMT`' with slots similar to those in '`ELT`'.

An '`ELT`' object also has the `optim` slot, which does not necessarily correspond to the EL optimization. The user can supply an arbitrary parameter value to test, reducing the problem to the EL evaluation. `elmt()` applies the single-step multiple testing procedure of Kim *et al.* (2021). The multiplicity-adjusted critical value and p values are estimated by Monte Carlo simulation.

Note that every step of the workflow involves possibly multiple EL evaluations or optimizations. Hence, it is necessary to flexibly control the details of the execution and computation at hand. All model fitting functions and most methods in accept an argument `control`, which allows the user to specify the control parameters. Only an object of class '`ControlEL`' can be supplied as `control` to ensure validity and avoid unexpected errors. Some of the slots in '`ControlEL`' are described in Table 2. An important feature is that '`ControlEL`' is independent of the other classes in the package, making it possible to apply different parameters for different tasks. Another wrapper, `el_control()`, is available to construct a '`ControlEL`' object and specify the parameters. The default values are shown below.

```
el_control(
  maxit = 200L, maxit_l = 25L, tol = 1e-06, tol_l = 1e-06, step = NULL,
  th = NULL, verbose = FALSE, keep_data = TRUE, nthreads,
  seed = sample.int(.Machine$integer.max, 1L), b = 10000L, m = 1000000L
)
```

Especially, `nthreads` specifies the number of threads for parallel computation via OpenMP (if available). By default, it is set to half the available threads and affects the following functions: `confint()`, `confreg()`, `el_lm()`, `el_glm()`, `eld()`, and `elt()`. For better performance, it is generally recommended in most platforms to limit the number of threads to the number of physical cores. `seed` sets the seed for random number generation. It defaults to a random integer generated from 1 to the maximum integer supported by R on the machine, which

Slot	Class	Description
`maxit`	`integer(1)`	Maximum number of iterations for the EL optimization.
`maxit_l`	`integer(1)`	Maximum number of iterations for the EL evaluation.
`tol`	`numeric(1)`	Convergence tolerance for the EL optimization.
`tol_l`	`numeric(1)`	Convergence tolerance for the EL evaluation.
`step`	`numeric(1)`	Step size for projected gradient descent method in the EL optimization.
`th`	`numeric(1)`	Threshold for the negative empirical log-likelihood ratio. The iteration stops if the value exceeds the threshold.
`nthreads`	`integer(1)`	Number of threads for parallel computation.

Table 2: A description of some of the slots in an '`ControlEL`' object. A full explanation of the class and slots can be found in the documentation of `ControlEL-class` or `el_control()` in the package.

is determined by `set.seed()`. For fast parallel random number generation and compatibility with OpenMP, the Xoshiro256+ pseudo-random number generator (period $2^{256} - 1$) of Blackman and Vigna (2021) is used internally with the **dqrng** package (Stubner 2021).

4. Usage

4.1. Model building

For a simple illustration of building a model, we apply `el_mean()` to the synthetic classification problem data `synth.tr` from the **MASS** package (Venables and Ripley 2002). The **dplyr** package (Wickham, François, Henry, and Müller 2022) and the **ggplot2** package (Wickham 2016) are used to aid data manipulation and visualization.

```
R> library("melt")
R> library("MASS")
R> library("dplyr")
R> library("ggplot2")
R> theme_set(theme_bw())
R> data("synth.tr", package = "MASS")
R> data <- dplyr::select(synth.tr, c(xs, ys))
R> summary(data)
```

```
       xs                  ys
 Min.   :-1.24652   Min.   :-0.1913
 1st Qu.:-0.50923   1st Qu.: 0.3234
 Median :-0.04183   Median : 0.4898
 Mean   :-0.07276   Mean   : 0.5044
 3rd Qu.: 0.36996   3rd Qu.: 0.7044
 Max.   : 0.86130   Max.   : 1.0932
```

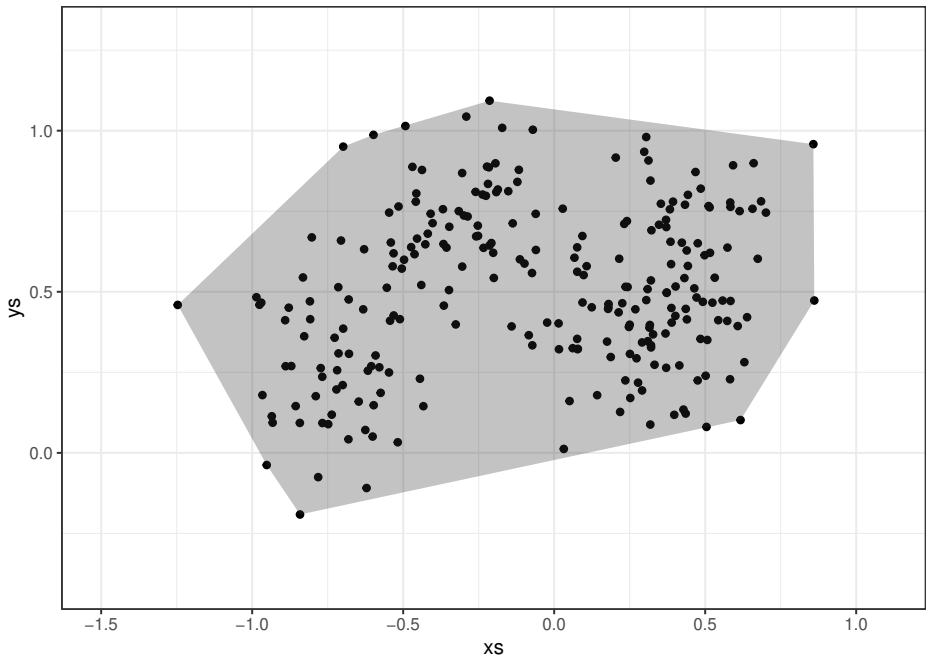

Figure 1: Scatter plot of `ys` versus `xs` in the `synth.tr` data with 250 observation. The convex hull of the observations is shaded in grey.

With the focus on `xs` and `ys`, the x and y coordinates, we first visualize the domain of the EL function with the convex hull constraint in Figure 1.

```
R> ggplot(data, aes(xs, ys)) +
+    geom_point() +
+    geom_polygon(data = slice(data, chull(xs, ys)), alpha = 0.3) +
+    xlim(-1.5, 1.1) +
+    ylim(-0.4, 1.3)
```

Any parameter value inside the convex hull leads to proper EL evaluation. We specify `c(0, 0.5)` as `par` in `el_mean()` and build an '`EL`' object with the '`data.frame`' `data`.

```
R> fit_mean <- el_mean(data, par = c(0, 0.5))
```

`data` is implicitly coerced into a '`matrix`' since `el_mean()` takes a numeric '`matrix`' as an input for the data. Basic `print` and `show` methods display relevant information about an '`EL`' object.

```
R> fit_mean

        Empirical Likelihood

Model: mean

Maximum EL estimates:
```

```
      xs       ys
-0.07276  0.50436

Chisq: 6.158, df: 2, Pr(>Chisq): 0.04601

EL evaluation: converged
```

The asymptotic chi-square statistic is displayed, along with the associated degrees of freedom and the p value. The MELE is just the average of the observations, and the empirical log-likelihood ratio is minimized at the MELE. We note that the MELE is independent of the `par` specified, which makes it convenient to build a model when the user is more interested in a subsequent analysis with an '`EL`' object.

```
R> fit2_mean <- el_mean(data, par = c(100, 100))
R> all.equal(coef(fit2_mean), colMeans(data))

[1] TRUE

R> fit3_mean <- el_mean(data, par = coef(fit2_mean))
R> all.equal(logLR(fit3_mean), 0)

[1] TRUE
```

As an illustration of weighted EL, we specify an arbitrary `weight` in `el_mean()` for weighted EL evaluation. The MELE is the weighted average of the observations in this case. The re-scaled weights returned by `weights()` add up to the total number of observations.

```
R> weights <- rep(c(1, 2), each = 125)
R> (wfit_mean <- el_mean(data, par = c(0, 0.5), weights = weights))

        Weighted Empirical Likelihood

Model: mean

Maximum EL estimates:
      xs       ys
-0.02319  0.56390

Chisq: 18.33, df: 2, Pr(>Chisq): 0.0001047

EL evaluation: converged

R> all.equal(sum(weights(wfit_mean)), nobs(wfit_mean))

[1] TRUE
```

Next, we consider an infeasible parameter value `c(1, 0,5)` outside the convex hull to show that how `el_control()` interacts with the model fitting functions through `control` argument. By employing the pseudo logarithm function in Equation 6, the evaluation algorithm continues until the iteration reaches `maxit_l` or the negative empirical log-likelihood ratio exceeds `th`. Setting a large `th` for the infeasible value, we observe that the algorithm hits the `maxit` with each element of `lambda` diverging quickly.

```
R> ctrl <- el_control(maxit_l = 50, th = 10000)
R> fit4_mean <- el_mean(data, par = c(1, 0.5), control = ctrl)
R> logL(fit4_mean)

[1] -10001.14

R> logLR(fit4_mean)

[1] -8620.776

R> getOptim(fit4_mean)

$par
 xs  ys
1.0 0.5

$lambda
[1] -9.908531e+14  2.757135e+14

$iterations
[1] 50

$convergence
[1] FALSE
```

We generate a surface plot of the empirical log-likelihood ratio on the grid of Figure 1. The boundary of the convex hull separates the feasible region from the infeasible region (Figure 2).

```
R> xs <- seq(-1.5, 1.1, length.out = 60)
R> ys <- seq(-0.4, 1.3, length.out = 40)
R> ctrl <- el_control(th = 400)
R> z <- matrix(NA_real_, nrow = length(xs), ncol = length(ys))
R> for (i in seq_len(length(xs))) {
+    for (j in seq_len(length(ys))) {
+      z[i, j] <- logLR(el_mean(data, par = c(xs[i], ys[j]), control = ctrl))
+    }
+  }
R> par(mar = c(1, 0, 0, 0))
R> persp(xs, ys, z,
+    xlab = "xs", ylab = "ys", zlab = "logLR", theta = 315,
+    phi = 25, d = 5, ticktype = "detailed"
+  )
```

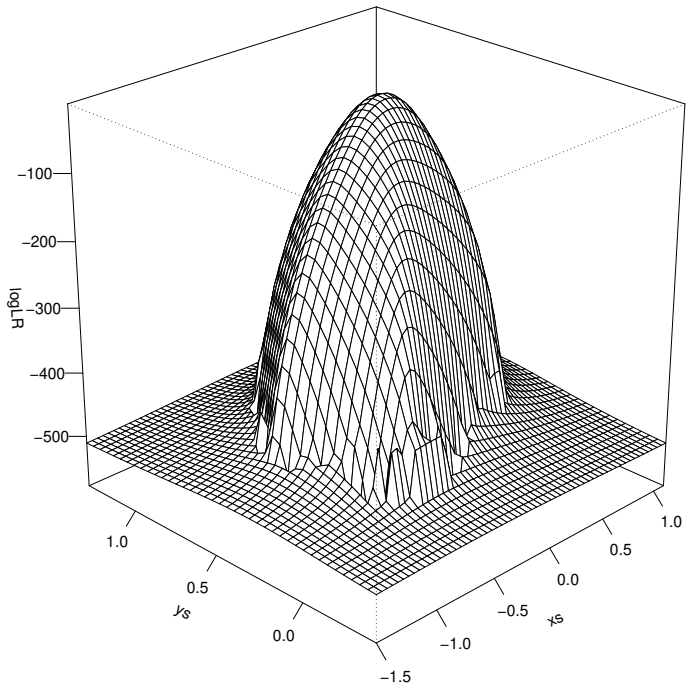

Figure 2: Surface plot of empirical log-likelihood ratio obtained from `synth.tr` with `el_mean()`. `th` is set to 400.

A similar process applies to the other model fitting functions, except that `el_lm()` and `el_glm()` require a '`formula`' object for model specification. In addition, **melt** contains another function `el_eval()` to perform the EL evaluation for other general estimating functions. For example, consider the mean and standard deviation denoted by $\theta = (\mu, \sigma)$. For a given value of θ, we evaluate the estimating function $g(X_i, \theta) = (X_i - \mu, (X_i - \mu)^2 - \sigma^2)$ with the available data $X_1, \ldots, X_n$. `el_eval()` takes a '`matrix`' argument `g`, where each row corresponds to $g(X_i, \theta)$.

```
R> mu <- 0
R> sigma <- 1
R> set.seed(123526)
R> x <- rnorm(100)
R> g <- matrix(c(x - mu, (x - mu)^2 - sigma^2), ncol = 2)
R> fit_eval <- el_eval(g)
R> fit_eval$pval

[1] 0.4645579
```

Although the user can supply a custom `g`, `el_eval()` is not the main function of the package. `el_eval()` returns a '`list`' with the same components as in an '`EL`' object, but no other methods are applicable further. The scope is also limited to just-identified estimating functions. For more flexible and over-identified estimating functions, it is recommended to use other packages, e.g., **gmm** or **momentfit**.

4.2. Linear regression analysis

We illustrate the use of `el_lm()` for regression analysis with the Boston housing price data `Boston` available in **MASS** (Venables and Ripley 2002). We first update the control parameters for significance tests of the coefficients.

```
R> data("Boston", package = "MASS")
R> ctrl <- el_control(maxit = 10000, tol = 1e-04, th = 10000, nthreads = 2)
R> (fit_lm <- el_lm(medv ~ crim + indus + chas + nox + age + lstat,
+    data = Boston,
+    control = ctrl
+  ))

        Empirical Likelihood

Model: lm

Maximum EL estimates:
(Intercept)        crim       indus        chas         nox
   32.76605    -0.05674    -0.17924     4.68855    -0.02926
        age       lstat
    0.04812    -0.91991

Chisq: 623.3, df: 6, Pr(>Chisq): < 2.2e-16

Constrained EL: converged
```

The `print()` method also applies and shows the MELE, the overall model test result, and the convergence status. The estimates are obtained from `lm.fit()`. The hypothesis for the overall test is that all the parameters except the intercept are 0. The convergence status shows that a constrained optimization is performed in testing the hypothesis. The EL evaluation applies to the test and the convergence status if the model does not include an intercept. `conv()` can be used to extract the convergence status.

```
R> conv(fit_lm)

[1] TRUE
```

It is designed to return a single logical, which can be helpful in a control flow where the convergence status decides the course of action. The large chi-square value implies that the data do not support the hypothesis, regardless of the convergence. Note that failure to converge does not necessarily indicate unreliable test results. Most commonly, the algorithm fails to converge if the additional constraint imposed by a hypothesis is incompatible with the convex hull constraint. The control parameters affect the test results as well. The `summary()` method reports the results of significance tests, where each test involves solving a constrained EL problem.

```
R> summary(fit_lm)
```

```
Call:
el_lm(formula = medv ~ crim + indus + chas + nox + age + lstat,
    data = Boston, control = ctrl)

Coefficients:
            Estimate   Chisq Pr(>Chisq)
(Intercept) 32.76605 385.205    < 2e-16 ***
crim        -0.05674   3.301   0.069223 .
indus       -0.17924  12.859   0.000336 ***
chas         4.68855  16.904   3.93e-05 ***
nox         -0.02926   0.001   0.973702
age          0.04812   9.936   0.001621 **
lstat       -0.91991 279.671    < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Chisq: 623.3, df: 6, Pr(>Chisq): < 2.2e-16

Constrained EL: converged
```

These tests are all asymptotically pivotal without explicit studentization. As a result, the output does not have standard errors. `sigTests()` returns the details of the tests.

By iteratively solving constrained EL problems for a grid of parameter values, confidence intervals for the parameters can be calculated with `confint()`. The chi-square calibration is the default, but the user can specify a critical value `cv` optionally. Below we calculate 90% confidence intervals with `ctrl`.

```
R> confint(fit_lm, level = 0.9, cv = NULL, control = ctrl)

                  lower        upper
(Intercept) 31.69350951 33.949400304
crim        -0.09398874 -0.006864787
indus       -0.25930773 -0.098861648
chas         2.70108821  6.987414019
nox         -2.76562281  2.941976149
age          0.02451114  0.071405941
lstat       -1.04020131 -0.800473217
```

Without standard errors and `vcov()` methods, the `lower` and `upper` confidence limits do not necessarily correspond to 5 and 95 percentiles, respectively. Similarly, we obtain confidence regions for two parameters with `confreg()`. Starting from the MELE, it computes the boundary points of a confidence region in full circle. An optional argument `npoints` controls the number of boundary points. The return value is a '`ConfregEL`' object containing a matrix whose rows consist of the points, and the `plot()` method visualizes the confidence region (Figure 3).

```
R> confreg <- confreg(fit_lm,
+    parm = c("crim", "lstat"), level = 0.9, cv = NULL,
```

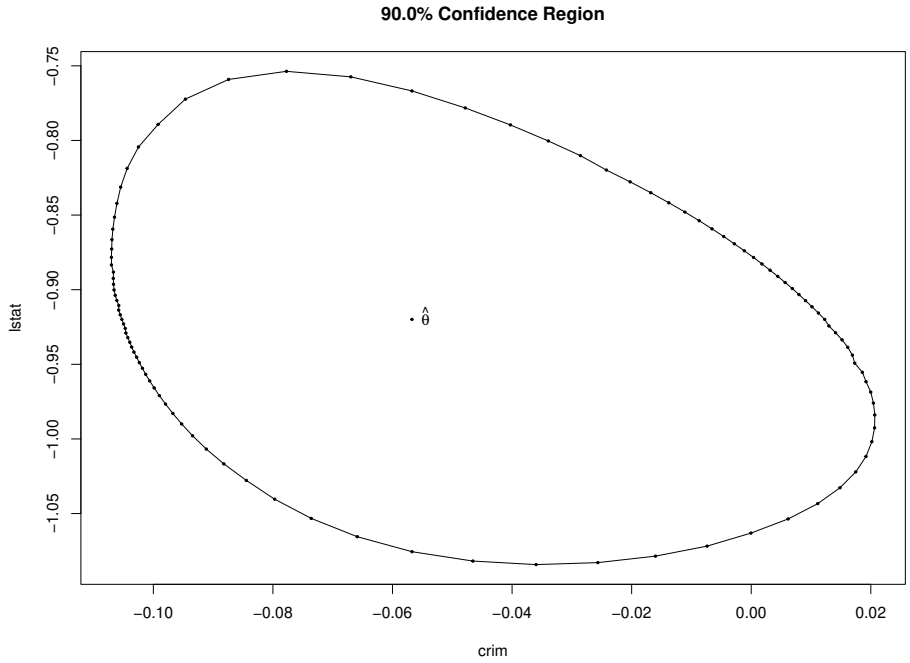

Figure 3: Scatter plot of the boundary points for asymptotic 90% confidence region of `crim` and `lstat` in `fit_lm`. $\hat{\theta}$ in the center of the plot is the MELE.

```
+    npoints = 100, control = ctrl
+  )
R> plot(cr)
```

Finally, we apply `eld()` to detect influential observations and outliers. Aside from the model object, `eld()` only accepts the control parameters. By the leave-one-out method of ELD, an '`ELD`' object inherits from the base type '`numeric`', with the length equal to the number of observations in the data. Figure 4 shows the ELD values from the `plot()` method.

```
R> eld <- eld(fit_lm, control = ctrl)
R> summary(eld)
R> plot(eld)

    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
  0.0000   0.3051   1.5199   7.5272   3.9676 199.8356
```

The code below shows that the observation with the largest ELD also has the largest Cook's distance from the same linear model fitted by `lm()`.

```
R> fit2_lm <- lm(medv ~ crim + indus + chas + nox + age + lstat,
+    data = Boston
+  )
R> cd <- cooks.distance(fit2_lm)
R> all.equal(which.max(eld), which.max(cd), check.attributes = FALSE)
```

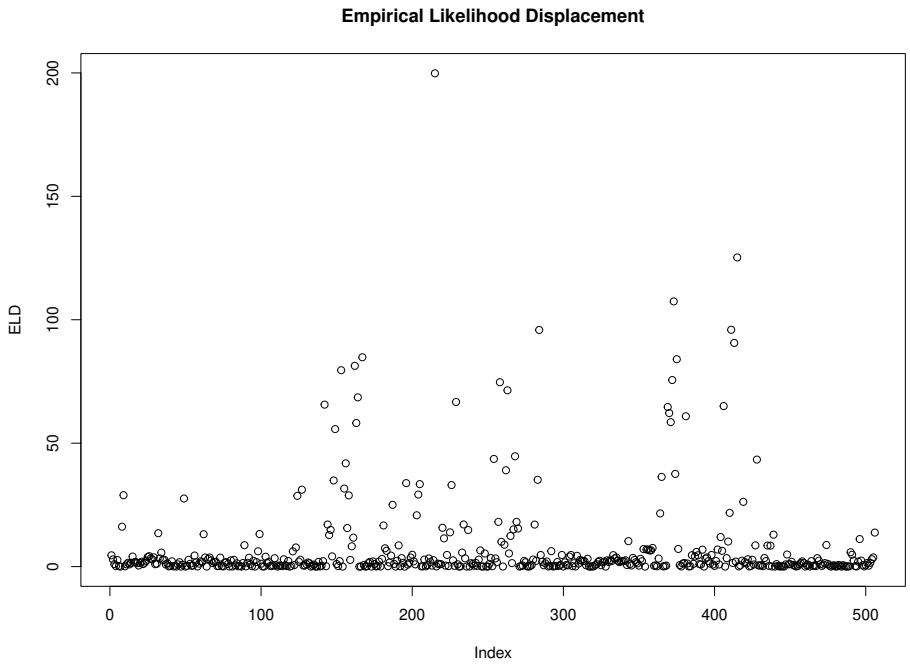

Figure 4: Scatter plot of empirical likelihood displacement versus observation index in `fit_lm`. The 215th observation has the largest value.

```
[1] TRUE
```

4.3. Hypothesis testing

Now we consider `elt()` for hypothesis testing, with the function prototype given below.

```
elt(object,
  rhs = NULL, lhs = NULL, alpha = 0.05, calibrate = "chisq",
  control = el_control()
)
```

Arguments `rhs` and `lhs` define a hypothesis and correspond to r and L in Equation 13, respectively. Therefore, either one of them must be provided. When `lhs` is `NULL`, it performs the EL evaluation at $\theta = r$ by setting $L = I_p$, where I_p is the identity matrix of order p. When `rhs` is `NULL`, on the other hand, r is set to the zero vector automatically, and the EL optimization is performed with L. Technically, `elt()` can reproduce all of the test results in the previous sections. Note the equivalence between the optimization results.

```
R> elt_mean <- elt(fit_mean, rhs = c(0, 0.5))
R> all.equal(getOptim(elt_mean), getOptim(fit_mean))
```

```
[1] TRUE
```

```
R> elt_lm <- elt(fit_lm, lhs = cbind(rep(0, 6), diag(6)), control = ctrl)
R> all.equal(getOptim(elt_lm), getOptim(fit_lm))

[1] TRUE
```

In addition to specifying an arbitrary linear hypothesis through `rhs` and `lhs`, extra arguments `alpha` and `calibrate` expand options for testing. `alpha` controls the significance level determining the critical value, and `calibrate` chooses the calibration method. `critVal()` extracts the critical value from an '`ELT`' object.

```
R> critVal(elt_mean)

[1] 5.991465
```

We apply the F and bootstrap calibrations to `fit_mean` at a significance level of 0.05. The number of threads is increased to 4 with 100000 bootstrap replicates in `el_control()`.

```
R> ctrl <- el_control(
+    maxit = 10000, tol = 1e-04, th = 10000, nthreads = 4, b = 100000
+  )
R> (elt_mean_f <- elt(fit_mean, rhs = c(0, 0.5), calibrate = "F"))

        Empirical Likelihood Test

Hypothesis:
xs = 0.0
ys = 0.5

Significance level: 0.05, Calibration: F

Statistic: 6.158, Critical value: 6.089

p-value: 0.04835

R> (elt_mean_boot <- elt(fit_mean,
+    rhs = c(0, 0.5), calibrate = "boot", control = ctrl
+  ))

        Empirical Likelihood Test

Hypothesis:
xs = 0.0
ys = 0.5

Significance level: 0.05, Calibration: Bootstrap

Statistic: 6.158, Critical value: 6.049

p-value: 0.04713
```

The above output shows that the F and bootstrap calibrations tend to produce slightly larger critical values than the chi-square calibration. These values can be used as the `cv` argument in `confint()` and `confreg()`, improving coverage probabilities when the sample size is small.

We next compare `elt()` with `lht()` in the **car** package (Fox and Weisberg 2019). For illustration, we fit a logistic regression model to the U.S. women's labor-force participation data `Mroz` from the **carData** package (Fox, Weisberg, and Price 2022) with `el_glm()` and `glm()`. We include all variables of `carData` in the model with the binary response variable `lfp`, which stands for labor-force participation. See the documentation of `carData` for a detailed description of the variables.

```
R> library("car")
R> data("Mroz", package = "carData")
R> fit_glm <- el_glm(lfp ~ .,
+    family = binomial(link = "logit"), data = Mroz, control = ctrl
+  )
R> fit2_glm <- glm(lfp ~ ., family = binomial(link = "logit"), data = Mroz)
```

Then we examine the results of the `confint()` and `summary()` methods. `confint.glm()` in the **MASS** package (Venables and Ripley 2002) computes the intervals for `fit2_glm`.

```
R> matrix(c(confint(fit_glm), confint(fit2_glm)),
+    ncol = 4, dimnames = list(
+      c(names(coef(fit2_glm))),
+      c("EL_lower", "EL_upper", "MASS_2.5%", "MASS_97.5%")
+    )
+  )

               EL_lower    EL_upper   MASS_2.5%  MASS_97.5%
(Intercept)  2.25618826  4.15681423  1.93697359  4.46630794
k5          -1.80187354 -1.14429048 -1.86089654 -1.08747196
k618        -0.18003267  0.05310788 -0.19839650  0.06867096
age         -0.07221601 -0.05360630 -0.08830325 -0.03813509
wcyes        0.41694955  1.21441844  0.36099360  1.26377557
hcyes       -0.23970750  0.47060701 -0.29200419  0.51679061
lwg          0.32613025  0.91809817  0.31402218  0.90697688
inc         -0.05020494 -0.01941354 -0.05099767 -0.01877093
```

```
R> summary(fit_glm)

Call:
el_glm(formula = lfp ~ ., family = binomial(link = "logit"),
    data = Mroz, control = ctrl)

Coefficients:
            Estimate  Chisq Pr(>Chisq)
(Intercept)  3.18214 36.144   1.83e-09 ***
k5          -1.46291 84.860    < 2e-16 ***
```

```
k618        -0.06457  1.174      0.279
age         -0.06287 30.460   3.41e-08 ***
wcyes        0.80727 18.261   1.93e-05 ***
hcyes        0.11173  0.390      0.532
lwg          0.60469 19.676   9.17e-06 ***
inc         -0.03445 22.983   1.63e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Dispersion for binomial family: 1

Chisq: 125.6, df: 7, Pr(>Chisq): < 2.2e-16

Constrained EL: converged
```

Based on the output above, we test two hypotheses that involve different classes of `lhs`: 1) `wc` = `hc` and 2) `k5` $= -1.5$ and `k618` $= 0$. Wald tests are performed by specifying `test = "Chisq"` in `lht()`.

```
R> lhs <- c(0, 0, 0, 0, 1, -1, 0, 0)
R> elt_glm <- elt(fit_glm, lhs = lhs, control = ctrl)
R> lht_glm <- lht(fit2_glm, hypothesis.matrix = lhs, test = "Chisq")
R> lhs2 <- rbind(
+    c(0, 1, 0, 0, 0, 0, 0, 0),
+    c(0, 0, 1, 0, 0, 0, 0, 0)
+  )
R> rhs2 <- c(-1.5, 0)
R> elt2_glm <- elt(fit_glm, rhs = rhs2, lhs = lhs2, control = ctrl)
R> lht2_glm <- lht(fit2_glm,
+    hypothesis.matrix = lhs2, rhs = rhs2, test = "Chisq"
+  )
```

For comparison, we extract the chi-square statistics and p values using `chisq()` and `pVal()`. The results are presented below.

```
R> matrix(c(
+    chisq(elt_glm), pVal(elt_glm),
+    lht_glm$Chisq[2], lht_glm$`Pr(>Chisq)`[2]
+  ),
+  nrow = 2, byrow = TRUE,
+  dimnames = list(c("EL", "Wald"), c("Chisq", "Pr(>Chisq)"))
+  )
```

```
        Chisq Pr(>Chisq)
EL   3.517759 0.06071449
Wald 3.536272 0.06004027
```

```
R> matrix(c(
+    chisq(elt2_glm), pVal(elt2_glm),
+    lht2_glm$Chisq[2], lht2_glm$`Pr(>Chisq)`[2]
+  ),
+  nrow = 2, byrow = TRUE,
+  dimnames = list(c("EL", "Wald"), c("Chisq", "Pr(>Chisq)"))
+  )

        Chisq Pr(>Chisq)
EL   1.144505  0.5642531
Wald 1.010919  0.6032282
```

The two tests provide similar results with a sample size of 753, which is not surprising given the asymptotic equivalence between these tests (see Qin and Lawless (1995) and references therein).

4.4. Multiple testing

We extend the hypothesis testing framework of Section 4.3 to multiple testing with `elmt()`. The syntax is similar to `elt()`, where `rhs` and `lhs` now specify multiple hypotheses.

```
elmt(object, rhs = NULL, lhs = NULL, alpha = 0.05, control = el_control())
```

`elmt()` employs a multivariate chi-square calibration technique based on Monte Carlo simulations to determine the common critical value. Details of multiple testing procedures are given in Kim *et al.* (2021). We use an internal dataset, `clothianidin`, on a pesticide concentration experiment by Alford and Krupke (2017). Clothianidin is a neonicotinoid pesticide widely applied to maize seeds. The original data are transformed into a simpler '`data.frame`'. We run a linear regression of `clo` on `trt` with `el_lm()`, where `clo` is a numeric vector of log-transformed clothianidin concentration and `trt` is a factor with four levels of different seed treatments.

```
R> data("clothianidin")
R> fit3_lm <- el_lm(clo ~ -1 + trt, data = clothianidin)
R> summary(fit3_lm)

Call:
el_lm(formula = clo ~ -1 + trt, data = clothianidin)

Coefficients:
             Estimate   Chisq Pr(>Chisq)
trtNaked       -4.479 411.072    < 2e-16 ***
trtFungicide   -3.427  59.486   1.23e-14 ***
trtLow         -2.800  62.955   2.11e-15 ***
trtHigh        -1.307   4.653      0.031 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Chisq: 894.4, df: 4, Pr(>Chisq): < 2.2e-16

EL evaluation: not converged
```

Each variable is significant, and the result of the model test shows that the seed treatments have an impact on the concentration.

Next, we perform all pairwise comparisons between the treatments to evaluate their differences. `elmt()` can be directly applied to the fitted model object with a contrast matrix for the comparisons.

```
R> contrast <- rbind(
+    c(1, -1, 0, 0), c(1, 0, -1, 0), c(1, 0, 0, -1),
+    c(0, 1, -1, 0), c(0, 1, 0, -1), c(0, 0, 1, -1)
+  )
R> elmt(fit3_lm, lhs = contrast)

        Empirical Likelihood Multiple Tests

Overall significance level: 0.05

Calibration: Multivariate chi-square

Hypotheses:
                             Estimate  Chisq Df   p.adj
trtNaked - trtFungicide = 0  -1.0525  5.510  1 0.08421 .
trtNaked - trtLow = 0        -1.6794 10.264  1 0.00709 **
trtNaked - trtHigh = 0       -3.1726 24.539  1 < 0.001 ***
trtFungicide - trtLow = 0    -0.6269  1.062  1 0.72458
trtFungicide - trtHigh = 0   -2.1201  8.397  1 0.01895 *
trtLow - trtHigh = 0         -1.4932  3.774  1 0.20409
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Common critical value: 6.513
```

The `Df` column shows the marginal chi-square degrees of freedom for each hypothesis. The matrix `contrast` is a particular type of `lhs` where each row corresponds to a hypothesis. For general hypotheses involving separate matrices, `elmt()` accepts '`list`' objects for `rhs` and `lhs`. The corresponding elements of `rhs` and `lhs` together form a hypothesis, as in Equation 13. We compare the result with the output of `glht()` in the **multcomp** package (Hothorn *et al.* 2008). `glht()` relies on (asymptotic) multivariate normal and t distributions for simultaneous tests.

```
R> library("multcomp")
R> fit4_lm <- lm(clo ~ -1 + trt, data = clothianidin)
R> summary(glht(fit4_lm, linfct = contrast))
```

```
         Simultaneous Tests for General Linear Hypotheses

Fit: lm(formula = clo ~ -1 + trt, data = clothianidin)

Linear Hypotheses:
       Estimate Std. Error t value Pr(>|t|)
1 == 0  -1.0525     0.6585  -1.598   0.3841
2 == 0  -1.6794     0.6882  -2.440   0.0761 .
3 == 0  -3.1726     0.6751  -4.699   <0.001 ***
4 == 0  -0.6269     0.6508  -0.963   0.7704
5 == 0  -2.1201     0.6370  -3.328   0.0065 **
6 == 0  -1.4932     0.6676  -2.237   0.1205
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Adjusted p values reported -- single-step method)
```

Based on the adjusted p values, both procedures reject the same hypotheses at an overall significance level of 0.05.

Finally, we use another internal dataset, `thiamethoxam`, from Obregon, Pederson, Taylor, and Poveda (2022) for an application to a more complex model. Like clothianidin, thiamethoxam is a neonicotinoid pesticide. Obregon *et al.* (2022) studied the effect of the thiamethoxam application method and plant variety on the number of bees visits. `thiamethoxam` is a '`data.frame`' with a variable `trt`, a factor with four levels of different application methods. Considering `visit`, bee visits per plot, as the response variable, we fit a quasi-Poisson regression model with a log link function using `el_glm()`.

```
R> data("thiamethoxam")
R> fit3_glm <- el_glm(visit ~ log(mass) + fruit + foliage + var + trt,
+    family = quasipoisson(link = "log"), data = thiamethoxam
+  )
R> summary(fit3_glm)

Call:
el_glm(formula = visit ~ log(mass) + fruit + foliage + var +
    trt, family = quasipoisson(link = "log"), data = thiamethoxam)

Coefficients:
             Estimate   Chisq Pr(>Chisq)
(Intercept)   0.74032 189.226    < 2e-16 ***
log(mass)     0.16938 401.439    < 2e-16 ***
fruit         0.04043  10.044    0.00153 **
foliage     -10.84203   5.293    0.02141 *
varGZ        -0.60770  35.914   2.06e-09 ***
trtSpray      0.06370   0.471    0.49259
trtFurrow    -0.04124   0.078    0.77939
trtSeed       0.14281   1.276    0.25857
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Dispersion estimate for quasipoisson family: 1.316692

Chisq: 117.2, df: 7, Pr(>Chisq): < 2.2e-16

Constrained EL: converged
```

The dispersion estimate is the MELE of ϕ in Equation 10. Thus it is smaller than the estimate returned by `summary()` when applied to a 'GLM' object. With no significant results found for `trt`, we test whether the coefficients are all 0. The hypothesis can also be specified symbolically using a character vector, which is suitable when there are many variables.

```
R> elt(fit3_glm, lhs = c("trtSpray", "trtFurrow", "trtSeed"))

        Empirical Likelihood Test

Hypothesis:
trtSpray = 0
trtFurrow = 0
trtSeed = 0

Significance level: 0.05, Calibration: Chi-square

Statistic: 1.504, Critical value: 7.815

p-value: 0.6813
```

We also test comparisons with control, and no significant differences are observed below.

```
R> elmt(fit3_glm, lhs = list("trtSpray", "trtFurrow", "trtSeed"))

        Empirical Likelihood Multiple Tests

Overall significance level: 0.05

Calibration: Multivariate chi-square

Hypotheses:
              Estimate Chisq Df p.adj
trtSpray = 0   0.06370 0.471  1 0.848
trtFurrow = 0 -0.04124 0.078  1 0.987
trtSeed = 0    0.14281 1.276  1 0.558

Common critical value: 5.611

R> fit4_glm <- glm(visit ~ log(mass) + fruit + foliage + var + trt,
+    family = quasipoisson, data = thiamethoxam
```

```
+  )
R> summary(glht(fit4_glm,
+    linfct = mcp(trt = c("Spray = 0", "Furrow = 0", "Seed = 0"))
+  ))

         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: User-defined Contrasts


Fit: glm(formula = visit ~ log(mass) + fruit + foliage + var + trt,
    family = quasipoisson, data = thiamethoxam)

Linear Hypotheses:
            Estimate Std. Error z value Pr(>|z|)
Spray == 0   0.06370    0.17870   0.356    0.971
Furrow == 0 -0.04124    0.18790  -0.219    0.993
Seed == 0    0.14281    0.17102   0.835    0.741
(Adjusted p values reported -- single-step method)
```

Note the use of a '`list`' for `lhs` by `elmt()`. While a character vector `lhs` acts as a single hypothesis for `elt()`, elements of `lhs` in `elmt()` define distinct hypotheses for convenience.

5. Conclusion

Empirical likelihood enables a likelihood-driven style of inference without the restrictive distributional assumptions of parametric models. Perhaps more importantly, while being nonparametric, empirical likelihood retains some desirable properties of parametric likelihood. In many ways, it is an attractive and natural approach to estimation and hypothesis testing, but its use has been limited due to computational difficulties compared to other methods. The R package **melt** aims to bridge the gap and provide a unified framework for data analysis with empirical likelihood methods. The package is developed to conduct statistical inference routinely made in R with empirical likelihood. Mainly, hypothesis testing is available for various models with smooth estimating functions. Examples in this paper demonstrate the functionality of **melt**. We provide more examples and details on the package website https://docs.ropensci.org/melt/. Future work will focus on expanding the scope to additional estimating functions and models. The package structure and its adoption of `S4` classes and methods are designed for extensibility. Optimization algorithms tailored to specific models can also be added in the process.

Acknowledgments

We thank Pierre Chausse and Alex Stringer for their comments and suggestions on the package during the rOpenSci review process. This work was supported by the U.S. National Science Foundation under Grants No. SES-1921523 and DMS-2015552.

References

Adimari G, Guolo A (2010). "A Note on the Asymptotic Behaviour of Empirical Likelihood Statistics." *Statistical Methods & Applications*, **19**(4), 463–476. doi:10.1007/s10260-010-0137-9.

Alford A, Krupke CH (2017). "Translocation of the Neonicotinoid Seed Treatment Clothianidin in Maize." *PLOS ONE*, **12**(3), 1–19. doi:10.1371/journal.pone.0173836.

Barton WH (2022). ***emplik2**: Empirical Likelihood Ratio Test for Two Samples with Censored Data*. R package version 1.32, URL https://CRAN.R-project.org/package=emplik2.

Bates D, Eddelbuettel D (2013). "Fast and Elegant Numerical Linear Algebra Using the **RcppEigen** Package." *Journal of Statistical Software*, **52**(5), 1–24. doi:10.18637/jss.v052.i05.

Benjamini Y, Hochberg Y (1995). "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing." *Journal of the Royal Statistical Society B*, **57**(1), 289–300. doi:10.1111/j.2517-6161.1995.tb02031.x.

Blackman D, Vigna S (2021). "Scrambled Linear Pseudorandom Number Generators." *ACM Trans. Math. Softw.*, **47**(4). ISSN 0098-3500. doi:10.1145/3460772. URL https://doi.org/10.1145/3460772.

Chambers JM (2014). "Object-Oriented Programming, Functional Programming and R." *Statistical Science*, **29**(2), 167–180. doi:10.1214/13-STS452.

Chaudhuri S, Mondal D, Yin T (2017). "Hamiltonian Monte Carlo Sampling in Bayesian Empirical Likelihood Computation." *Journal of the Royal Statistical Society B*, **79**(1), 293–320. doi:https://doi.org/10.1111/rssb.12164.

Chaussé P (2010). "Computing Generalized Method of Moments and Generalized Empirical Likelihood with R." *Journal of Statistical Software*, **34**(11), 1–35. doi:10.18637/jss.v034.i11.

Chaussé P (2022). ***momentfit**: Methods of Moments*. R package version 0.3, URL https://CRAN.R-project.org/package=momentfit.

Chen SX, Cui H (2003). "An Extended Empirical Likelihood for Generalized Linear Models." *Statistica Sinica*, **13**(1), 69–81.

Chen SX, van Keilegom I (2009). "A Review on Empirical Likelihood Methods for Regression." *Test*, **18**(3), 415–447. doi:10.1007/s11749-009-0159-5.

Chen SX, Peng L, Qin YL (2009). "Effects of Data Dimension on Empirical Likelihood." *Biometrika*, **96**(3), 711–722. doi:10.1093/biomet/asp037.

Cook RD (1986). "Assessment of Local Influence." *Journal of the Royal Statistical Society B*, **48**(2), 133–155. doi:10.1111/j.2517-6161.1986.tb01398.x.

Dagum L, Menon R (1998). "OpenMP: An Industry Standard API for Shared-Memory Programming." *IEEE Computational Science and Engineering*, **5**(1), 46–55. doi:10.1109/99.660313.

DiCiccio T, Hall P, Romano J (1991). "Empirical Likelihood is Bartlett-Correctable." *The Annals of Statistics*, **19**(2), 1053–1061. doi:10.1214/aos/1176348137.

DiCiccio TJ, Romano JP (1990). "Nonparametric Confidence Limits by Resampling Methods and Least Favorable Families." *International Statistical Review / Revue Internationale de Statistique*, **58**(1), 59–76. doi:10.2307/1403474.

Dickhaus T, Royen T (2015). "A Survey on Multivariate Chi-Square Distributions and Their Applications in Testing Multiple Hypotheses." *Statistics*, **49**(2), 427–454. doi:10.1080/02331888.2014.993639.

Dickhaus T, Sirotko-Sibirskaya N (2019). "Simultaneous Statistical Inference in Dynamic Factor Models: Chi-Square Approximation and Model-Based Bootstrap." *Computational Statistics & Data Analysis*, **129**, 30–46. doi:10.1016/j.csda.2018.08.012.

Eddelbuettel D, Balamuta JJ (2018). "Extending R with C++: A Brief Introduction to **Rcpp**." *The American Statistician*, **72**(1), 28–36. doi:10.1080/00031305.2017.1375990.

Efron B (1981). "Nonparametric Standard Errors and Confidence Intervals." *Canadian Journal of Statistics*, **9**(2), 139–158. doi:10.2307/3314608.

Fox J, Weisberg S (2019). *An R Companion to Applied Regression*. Third edition. Sage, Thousand Oaks CA. URL https://socialsciences.mcmaster.ca/jfox/Books/Companion/.

Fox J, Weisberg S, Price B (2022). ***carData**: Companion to Applied Regression Data Sets*. R package version 3.0-5, URL https://CRAN.R-project.org/package=carData.

Glenn N, Zhao Y (2007). "Weighted Empirical Likelihood Estimates and Their Robustness Properties." *Computational Statistics & Data Analysis*, **51**(10), 5130–5141. doi:10.1016/j.csda.2006.07.032.

Hall P, Scala BL (1990). "Methodology and Algorithms of Empirical Likelihood." *International Statistical Review / Revue Internationale de Statistique*, **58**(2), 109–127. doi:10.2307/1403462.

Hansen LP (1982). "Large Sample Properties of Generalized Method of Moments Estimators." *Econometrica*, **50**(4), 1029–1054. doi:10.2307/1912775.

Hansen LP, Heaton J, Yaron A (1996). "Finite-Sample Properties of Some Alternative GMM Estimators." *Journal of Business & Economic Statistics*, **14**(3), 262–280. doi:10.2307/1392442.

Hjort NL, McKeague IW, van Keilegom I (2009). "Extending the Scope of Empirical Likelihood." *The Annals of Statistics*, **37**(3), 1079–1111. doi:10.1214/07-AOS555.

Hothorn T, Bretz F, Westfall P (2008). "Simultaneous Inference in General Parametric Models." *Biometrical Journal*, **50**(3), 346–363. doi:10.1002/bimj.200810425.

Imbens GW (1997). "One-Step Estimators for Over-Identified Generalized Method of Moments Models." *The Review of Economic Studies*, **64**(3), 359–383. doi:10.2307/2971718.

Jacod J, Sørensen M (2018). "A Review of Asymptotic Theory of Estimating Functions." *Statistical Inference for Stochastic Processes*, **21**(2), 415–434. doi:10.1007/s11203-018-9178-8.

Kien DT, Chaudhuri S, Wei NH (2017). ***elhmc**: Sampling from a Empirical Likelihood Bayesian Posterior of Parameters Using Hamiltonian Monte Carlo*. R package version 1.1.0, URL https://CRAN.R-project.org/package=elhmc.

Kim E (2022). ***melt**: Multiple Empirical Likelihood Tests*. R package version 1.8.0, URL https://CRAN.R-project.org/package=melt.

Kim E, MacEachern S, Peruggia M (2021). "Empirical Likelihood for the Analysis of Experimental Designs." arXiv:2112.09206 [stat.ME]. URL https://arxiv.org/abs/2112.09206.

Kitamura Y (1997). "Empirical Likelihood Methods with Weakly Dependent Processes." *The Annals of Statistics*, **25**(5), 2084–2102. doi:10.1214/aos/1069362388.

Kitamura Y, Stutzer M (1997). "An Information-Theoretic Alternative to Generalized Method of Moments Estimation." *Econometrica*, **65**(4), 861–874. doi:10.2307/2171942.

Kitamura Y, Tripathi G, Ahn H (2004). "Empirical Likelihood-Based Inference in Conditional Moment Restriction Models." *Econometrica*, **72**(6), 1667–1714. doi:10.1111/j.1468-0262.2004.00550.x.

Kolaczyk ED (1994). "Empirical Likelihood for Generalized Linear Models." *Statistica Sinica*, **4**(1), 199–218.

Lazar NA (2005). "Assessing the Effect of Individual Data Points on Inference From Empirical Likelihood." *Journal of Computational and Graphical Statistics*, **14**(3), 626–642. doi:10.1198/106186005X59568.

Li G, Li R, Zhou M (2005). *Empirical Likelihood in Survival Analysis*, pp. 337–349. World Scientific. doi:10.1142/9789812567765_0020.

Newey WK, Smith RJ (2004). "Higher Order Properties of GMM and Generalized Empirical Likelihood Estimators." *Econometrica*, **72**(1), 219–255. doi:10.1111/j.1468-0262.2004.00482.x.

Nordman DJ, Lahiri SN (2014). "A Review of Empirical Likelihood Methods for Time Series." *Journal of Statistical Planning and Inference*, **155**, 1–18. doi:10.1016/j.jspi.2013.10.001.

Obregon D, Pederson G, Taylor A, Poveda K (2022). "The Pest Control and Pollinator Protection Dilemma: The Case of Thiamethoxam Prophylactic Applications in Squash Crops." *PLOS ONE*, **17**(5), 1–18. doi:10.1371/journal.pone.0267984.

Owen A (1988). "Empirical Likelihood Ratio Confidence Intervals for a Single Functional." *Biometrika*, **75**(2), 237–249. doi:10.1093/biomet/75.2.237.

Owen A (1990). "Empirical Likelihood Ratio Confidence Regions." *The Annals of Statistics*, **18**(1), 90–120. doi:10.1214/aos/1176347494.

Owen A (1991). "Empirical Likelihood for Linear Models." *The Annals of Statistics*, **19**(4), 1725–1747. doi:10.1214/aos/1176348368.

Owen A (2001). *Empirical Likelihood*. Chapman & Hall/CRC. doi:10.1201/9781420036152.

Qin J, Lawless J (1994). "Empirical Likelihood and General Estimating Equations." *The Annals of Statistics*, **22**(1), 300–325. doi:10.1214/aos/1176325370.

Qin J, Lawless J (1995). "Estimating Equations, Empirical Likelihood and Constraints on Parameters." *Canadian Journal of Statistics*, **23**(2), 145–159. doi:10.2307/3315441.

R Core Team (2022). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Shen B, Wang M (2022). ***ELCIC**: The Empirical Likelihood-Based Consistent Information Criterion*. R package version 0.2.0, URL https://CRAN.R-project.org/package=ELCIC.

Smith RJ (1997). "Alternative Semi-parametric Likelihood Approaches to Generalised Method of Moments Estimation." *The Economic Journal*, **107**(441), 503–519. doi:10.1111/j.0013-0133.1997.174.x.

Stein C (1956). "Efficient Nonparametric Testing and Estimation." In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pp. 187–195. doi:10.1525/9780520313880.

Stubner R (2021). ***dqrng**: Fast Pseudo Random Number Generators*. R package version 0.3.0, URL https://CRAN.R-project.org/package=dqrng.

Tang CY, Wu TT (2014). "Nested Coordinate Descent Algorithms for Empirical Likelihood." *Journal of Statistical Computation and Simulation*, **84**(9), 1917–1930. doi:10.1080/00949655.2013.770514.

Tsao M, Wu F (2013). "Empirical Likelihood on the Full Parameter Space." *The Annals of Statistics*, **41**(4), 2176–2196. doi:10.1214/13-AOS1143.

Valeinis J, Cers E (2022). ***EL**: Two-Sample Empirical Likelihood*. R package version 1.1, URL https://CRAN.R-project.org/package=EL.

Venables WN, Ripley BD (2002). *Modern Applied Statistics with S*. Fourth edition. Springer, New York. ISBN 0-387-95457-0, URL https://www.stats.ox.ac.uk/pub/MASS4/.

Wang L, Yang D (2018). "F-Distribution Calibrated Empirical Likelihood Ratio Tests for Multiple Hypothesis Testing." *Journal of Nonparametric Statistics*, **30**(3), 662–679. doi:10.1080/10485252.2018.1461867.

Wedderburn RWM (1974). "Quasi-Likelihood Functions, Generalized Linear Models, and the Gauss—Newton Method." *Biometrika*, **61**(3), 439–447. doi:10.1093/biomet/61.3.439.

Wickham H (2016). ***ggplot2**: Elegant Graphics for Data Analysis*. Springer-Verlag. ISBN 978-3-319-24277-4. URL https://ggplot2.tidyverse.org.

Wickham H, François R, Henry L, Müller K (2022). ***dplyr**: A Grammar of Data Manipulation*. R R package version 1.0.9, URL https://CRAN.R-project.org/package=dplyr.

Wu F, Zhang Y (2015). ***eel**: Extended Empirical Likelihood*. R package version 1.1, URL https://CRAN.R-project.org/package=eel.

Yuan KH, Jennrich RI (1998). "Asymptotics of Estimating Equations under Natural Conditions." *Journal of Multivariate Analysis*, **65**(2), 245–260. doi:10.1006/jmva.1997.1731.

Zhou M (2015). *Empirical Likelihood Method in Survival Analysis*, volume 79. Chapman & Hall/CRC. doi:10.1201/b18598.

Zhou M (2022). ***emplik**: Empirical Likelihood Ratio for Censored/Truncated Data*. R package version 1.2, URL https://CRAN.R-project.org/package=emplik.

Zhu H, Ibrahim JG, Tang N, Zhang H (2008). "Diagnostic Measures for Empirical Likelihood of General Estimating Equations." *Biometrika*, **95**(2), 489–507. doi:10.1093/biomet/asm094.

Affiliation:

Eunseop Kim
Department of Statistics
The Ohio State University
1958 Neil Ave.
Columbus, OH 43210, United States of America
E-mail: kim.7302@osu.edu

Steven N. MacEachern
Department of Statistics
The Ohio State University
1958 Neil Ave.
Columbus, OH 43210, United States of America
E-mail: snm@stat.osu.edu

Mario Peruggia
Department of Statistics
The Ohio State University
1958 Neil Ave.
Columbus, OH 43210, United States of America
E-mail: peruggia@stat.osu.edu