The Stata Journal (*yyyy*) *vv*, Number *ii*, pp. 1–40

Weighted-average least squares: Beyond the classical linear regression model

Giuseppe De Luca
University of Palermo
Palermo, Italy
giuseppe.deluca@unipa.it

Jan R. Magnus
Vrije Universiteit Amsterdam
Amsterdam, The Netherlands
jan@janmagnus.nl

Abstract. This paper introduces four new commands for the weighted-average least squares approach to model uncertainty: the `hetwals` command fits linear models with multiplicative forms of heteroskedasticity; the `ar1wals` command fits linear models with stationary AR(1) errors; the `xtwals` command fits fixed-effects and random-effects panel-data models with either i.i.d. or AR(1) idiosyncratic errors; while the `glmwals` command fits univariate generalized linear models. These commands extend the new functionalities of the `wals` command (version 3.0) introduced by De Luca and Magnus (2024), and enlarge the classes of models that can be fitted by this model-averaging method. We also provide an illustration of the `hetwals` and `glmwals` commands by means of real data applications.

Keywords: st0001, `hetwals`, `ar1wals`, `xtwals`, `glmwals`, `predict`, weighted-average least squares, heteroskedasticity, serial correlation, panel data, generalized linear models

1 Introduction

Weighted-average least squares (WALS) is a powerful and computationally efficient model-averaging technique, which accounts for both the estimation and the model-selection noise in one joint procedure. In De Luca and Magnus (2024) we introduced version 3.0 of the `wals` command, which implements the WALS estimator of Magnus et al. (2010) for linear models with independent and identically distributed (i.i.d.) errors. The present paper, which is a sequel of the previous one, introduces four new commands of the WALS package in Stata that considerably enlarge the classes of models that can be fitted by this model-averaging method: the `hetwals` command fits linear models with multiplicative forms of heteroskedasticity; the `ar1wals` command fits linear models with stationary AR(1) errors; the `xtwals` command fits fixed-effects and random-effects panel-data models with either i.i.d. or AR(1) idiosyncratic errors; while the `glmwals` command fits univariate generalized linear models (GLMs) such as gamma, logit, probit, complementary log-log, and Poisson regressions.

As discussed in Magnus and De Luca (2016), all these generalizations rely on simple methods of transformation or linearization coming from classical least-squares (LS) theory. The WALS approach to linear models with nonspherical errors was developed by Magnus et al. (2011), who proposed weakening the assumption of i.i.d. errors using a general parametric model for the variance matrix. The linear model with multiplicative

forms of heteroskedasticity, the linear model with AR(1) errors, and the random-effects approach to panel-data models represent special cases of this more general parametric approach. In these cases, the estimation procedure is based on a feasible generalized least squares (FGLS) strategy where one first estimates the unknown parameters of the variance matrix from the unrestricted model and then performs WALS on the FGLS transformations of the outcome variable and the regressors. The fixed-effects approach to panel-data models is motivated by the fact that the WALS estimator still satisfies the Frisch–Waugh–Lovell theorem when the individual effects are treated as (nuisance) focus parameters. Thus, as with the classical fixed-effects estimator, one can perform WALS on the familiar within-transformations of the data to wipe out the individual effects. `xtwals` also supports the fixed-effects and random-effects WALS estimators for panel-data models with AR(1) errors using the same developments of the official `xtregar` command. The WALS approach to GLMs was developed by De Luca et al. (2018), who proposed linearizing the nonlinear system of likelihood equations as in the Newton–Raphson and Fisher scoring algorithms. The main difference with respect to the classical iteratively reweighted least squares (IRLS) procedure is that, instead of iterating the optimization until convergence, one performs WALS on the data transformations obtained in the first iteration. Based on the theory of one-step maximum likelihood (ML) estimators, a single iteration of the estimation procedure leads to the one-step WALS estimator. To weaken the dependence of this estimator on the starting values, the default estimation method implemented by `glmwals` is an iterative procedure that repeatedly updates the starting values using the one-step WALS estimates from the previous iteration until some convergence criterion is satisfied.

The remainder of the paper is organized as follows. Section 2 discusses the WALS approach to linear models with nonspherical errors, fixed-effects and random-effects panel-data models, and GLMs. Syntax, options, and predictions of the new Stata commands are described in the four subsequent sections: `hetwals` in Section 3, `ar1wals` in Section 4, `xtwals` in Section 5, and `glmwals` in Section 6. Sections 7 and 8 present two empirical applications: the `hetwals` estimates of a hedonic housing price model for the Hong Kong residential property market, and the `glmwals` estimates of binary choice models for the attrition process in the first two waves of the Survey of Health, Aging and Retirement in Europe (SHARE). Additional examples on the new WALS commands are presented in the on-line help files. Section 9 concludes. The results stored by the new WALS commands are listed in Appendix A.

2 Extensions of the WALS approach

The basic setup of WALS is the partitioned homoskedastic linear model

$$y = X_1\beta_1 + X_2\beta_2 + \epsilon, \tag{1}$$

where y $(n\times 1)$ is the vector of observations on the outcome, X_1 $(n\times k_1)$ and X_2 $(n\times k_2)$ are matrices of nonrandom regressors, β_1 and β_2 are the associated parameter vectors, and ϵ is a vector of random errors. We assume that $k_1 \geq 0$, $k_2 \geq 1$, $X = (X_1, X_2)$ has full column-rank $k = k_1 + k_2 < n$, and that the errors are i.i.d. $\mathrm{N}(0, \sigma^2_\epsilon I_n)$, where

$0 < \sigma_\epsilon^2 < \infty$ and I_n is the identity matrix of order n.[1] The reason for partitioning X in two blocks is that X_1 contains the *focus* regressors that we want to include with certainty in the model, while X_2 contains the *auxiliary* regressors of which we are less certain. Since each of the k_2 auxiliary regressors can be either included or not, we have 2^{k_2} models to consider.

As discussed in Magnus et al. (2010), Magnus et al. (2016), Magnus and De Luca (2016), De Luca et al. (2022, 2023, 2025), and De Luca and Magnus (2024), WALS is a frequentist model-averaging method that builds on a semi-orthogonal transformation of the auxiliary regressors and a Bayesian analysis of the normal location model resulting from this preliminary step. In particular, the semi-orthogonal transformation produces k_2 unique linear combinations of the auxiliary regressors such that the residuals of their linear projections on the focus regressors are independent of each other. This step allows us to reduce the computational burden of the model-averaging estimator of the transformed parameters from order 2^{k_2} to order k_2. Moreover, it allows the development of a Bayesian weighting scheme that ensures a proper treatment of prior ignorance and other desirable theoretical properties, in particular admissibility, bounded risk, minimax regret optimality, and robustness. Given the model-averaging estimates of the transformed parameters, the original parameters are estimated using their one-to-one relationships with the transformed parameters. In what follows, we describe three extensions of this model-averaging method. We shall assume that the reader is familiar with the standard WALS approach to model (1) and the recent version 3.0 of the `wals` command developed by De Luca and Magnus (2024).

2.1 Linear models with nonspherical errors

The assumption of i.i.d. errors in model (1) is essential for the independence between the LS estimator of β_1 in the fully restricted model with $\beta_2 = 0$ and the LS estimator of β_2 in the unrestricted model (Magnus and De Luca 2016, Proposition 2.1), which in turn is an essential ingredient of the WALS approach. Although this assumption cannot be easily relaxed, it can be weakened by assuming a linear model of the form:

$$y = X_1\beta_1 + X_2\beta_2 + u, \qquad u \sim \mathrm{N}(0, \Omega), \tag{2}$$

where $\Omega = \sigma_\epsilon^2 V$ and $V = V(\alpha)$ is a positive definite $n \times n$ matrix whose elements are known smooth functions of a p-dimensional parameter vector $\alpha = (\alpha_1, \ldots, \alpha_p)'$. The WALS approach for this more general setup employs a FGLS strategy that mimics the case where only σ_ϵ^2 needs to be estimated. Specifically, if V were known, we would perform WALS on the transformed model

$$y^* = X_1^*\beta_1 + X_2^*\beta_2 + \epsilon, \tag{3}$$

1. The assumption of nonrandom regressors plays no role in the development of the WALS theory and is used only to suppress the conditioning on X. If we think of X as being random, then it must be assumed that $\epsilon|X \sim \mathrm{N}(0, \sigma_\epsilon^2 I_n)$ and that the determinant of $X'X$ is positive with probability one. Similarly, the normality assumption is only needed to justify the estimation procedure and to derive the finite-sample distribution of the WALS estimator, but it can be relaxed in the asymptotic theory under the usual conditions of the central limit theorem.

where $y^* = V^{-1/2}y$, $X_1^* = V^{-1/2}X_1$, $X_2^* = V^{-1/2}X_2$, and $\epsilon = V^{-1/2}u \sim \mathrm{N}(0, \sigma_\epsilon^2 I_n)$. Since V is not known, we first compute an estimate $\hat{\alpha}$ of its parameters α from the unrestricted model, through which we obtain a plug-in estimate $\hat{V} = V(\hat{\alpha})$ of V. In the second step of the procedure, we ignore the randomness in the estimation of α, and estimate β_1, β_2, and σ_ϵ^2 by a standard WALS regression of $\hat{y}^* = \hat{V}^{-1/2}y$ on $\hat{X}_1^* = \hat{V}^{-1/2}X_1$ and $\hat{X}_2^* = \hat{V}^{-1/2}X_2$. Even though theoretical results on the influence of this neglected randomness are not yet available, the Monte Carlo (MC) simulations of Magnus et al. (2011) and Magnus et al. (2016) suggest that this FGLS strategy may substantially improve the mean squared error (MSE) performance of WALS point estimates and WALS predictions.

Sections 3 and 4 introduce the WALS commands designed for two special cases of model (2): `hetwals` is designed for linear models with multiplicative forms of heteroskedasticity (Harvey 1976), while `ar1wals` is designed for linear models with AR(1) errors (Davidson and MacKinnon 1993, Chapter 10).

The model with multiplicative heteroskedasticity assumes that $\Omega = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_n^2)$, where $\sigma_i^2 = \exp(\alpha' v_i)$ and $v_i = (1, v_{2i}, \ldots, v_{pi})'$ is a vector of $p \geq 2$ regressors for the variance function, which includes a constant term and an additional subset of variables that are meant to capture departures from homoskedasticity. When $\alpha_2 = \cdots = \alpha_p = 0$, we obtain the basic setup with homoskedastic errors where $\sigma_i^2 = \exp(\alpha_1) = \sigma_\epsilon^2$. Based on this setup, we first apply the `hetregress` command to the unrestricted model to estimate the parameters α by either ML or Harvey's two-step GLS method. In the second step, we exploit the fact that Ω is a diagonal matrix to fit a weighted WALS regression of y on X_1 and X_2 with analytic weights equal to the reciprocals of the estimated variances $\hat{\sigma}_i^2 = \exp(\hat{\alpha}' v_i)$.

The model with stationary AR(1) errors represents the simplest and most common autocorrelated error process used in the analysis of time-series data. Indexing the random errors by the subscript t, we have

$$u_t = \rho u_{t-1} + \epsilon_t, \tag{4}$$

where $|\rho| < 1$ is the correlation coefficient and ϵ_t are i.i.d. errors that follow a $\mathrm{N}(0, \sigma_\epsilon^2)$ distribution. In terms of model (2), this corresponds to setting

$$V = \frac{1}{1-\rho^2} \begin{pmatrix} 1 & \rho & \rho^2 & \ldots & \rho^{n-1} \\ \rho & 1 & \rho & \ldots & \rho^{n-2} \\ \rho^2 & \rho & 1 & \ldots & \rho^{n-3} \\ \vdots & \vdots & \vdots & & \vdots \\ \rho^{n-1} & \rho^{n-2} & \rho^{n-3} & \ldots & 1 \end{pmatrix}. \tag{5}$$

When $\rho = 0$, we obtain the basic setup with i.i.d. errors. Under model (4), we first estimate ρ from the LS residuals of the unrestricted model through one of the six methods available in the `prais` command (see Section 4). Then, we estimate a WALS regression based on the Cochrane–Orcutt or Prais–Winsten transformations of the outcome and the regressors, which transform the errors into white noise.

2.2 Linear and static panel-data models

Consider next a panel dataset containing repeated observations over time on a set of statistical units. In this case, our statistical framework is the following generalization of model (1):

$$y_{it} = \nu_i + x'_{it,1}\beta_1 + x'_{it,2}\beta_2 + u_{it}, \qquad i = 1, \ldots, N, \quad t = 1, \ldots, T, \tag{6}$$

where $(y_{it}, x'_{it,1}, x'_{it,2})$ are the observations on the outcome of interest y_{it}, the focus regressors $x_{it,1}$, and the auxiliary regressors $x_{it,2}$ for the ith unit in the tth period; and the ν_i are the individual effects that are meant to capture unobserved and time-invariant heterogeneity. The total number of observations is thus $n = NT$.

We consider two specifications of model (6): the basic setup and the extended setup. The extended setup assumes that

$$u_{it} = \rho u_{i,t-1} + \epsilon_{it}, \tag{7}$$

where $|\rho| < 1$ and the ϵ_{it} are i.i.d. errors that follow a $\mathrm{N}(0, \sigma_\epsilon^2)$ distribution. Setting $\rho = 0$ gives the basic setup with i.i.d. errors. In either case, the panel is allowed to be (exogenously) unbalanced and unequally spaced over time, but this is not made explicit for ease of notation. The assumption of nonrandom regressors now amounts to assuming strict exogeneity of the regressors, that is, the ϵ_{it} must be mean independent of past, present, and future values of $x_{it} = (x'_{it,1}, x'_{it,2})'$. This condition rules out dynamic panel-data models where lags of the outcome variable are treated as predetermined regressors.

In vector form, model (6) can be written more compactly as

$$y_i = \imath_T \nu_i + X_{i,1}\beta_1 + X_{i,2}\beta_2 + u_i, \qquad i = 1, \ldots, N, \tag{8}$$

where $\imath_T$ denotes the $T \times 1$ vector with elements all equal to one, and

$$y_i = \begin{pmatrix} y_{i1} \\ \vdots \\ y_{iT} \end{pmatrix}, \qquad X_{i,1} = \begin{pmatrix} x'_{i1,1} \\ \vdots \\ x'_{iT,1} \end{pmatrix}, \qquad X_{i,2} = \begin{pmatrix} x'_{i1,2} \\ \vdots \\ x'_{iT,2} \end{pmatrix}, \qquad u_i = \begin{pmatrix} u_{i1} \\ \vdots \\ u_{iT} \end{pmatrix}.$$

Letting $y = (y_1, \ldots, y_N)'$, $D = I_N \otimes \imath_T$, $\nu = (\nu_1, \ldots, \nu_N)'$, $X_1 = (X'_{1,1}, \ldots, X'_{N,1})'$, $X_2 = (X'_{1,2}, \ldots, X'_{N,2})'$, and $u = (u_1, \ldots, u_N)'$, our model can be written even more compactly as

$$y = D\nu + X_1\beta_1 + X_2\beta_2 + u. \tag{9}$$

There are two classical approaches to the estimation of model (9), which differ depending on the assumptions made on the individual effects ν_i (see, e.g., Mundlak 1978). In the fixed-effects approach, we treat the ν_i as an additional subset of (nuisance) focus parameters to be estimated jointly with the other parameters of the model.[2] The fixed-effects approach is motivated by the fact that unobserved time-invariant heterogeneity

2. As usual, in the fixed-effects approach, $X = (X_1, X_2)$ cannot include time-invariant regressors. Moreover, we impose the restriction $\sum_i \nu_i = 0$ to avoid perfect collinearity with the constant term.

is often a relevant source of omitted-variable bias. Thus, the alternative strategies of omitting the matrix D of unit-specific dummy variables or treating D as an additional subset of auxiliary regressors together with X_2 are unlikely to produce better estimators of β_1 and β_2 in the MSE sense. The key issue is that, when N is sufficiently large, brute force estimation of a WALS regression of y on $\tilde{X}_1 = (X_1, D)$ and X_2 may not be feasible as it would require inverting the matrix $\tilde{X}_1'\tilde{X}_1$. In contrast, the random-effects approach treats the ν_i as i.i.d. random variables that (conditionally on x_{it}) follow a $\mathrm{N}(0, \sigma_\nu^2)$ distribution. The ν_i are also assumed to be independent of the u_{it} for all i and all t. These conditions lead to the well-known variance component model

$$y = X_1\beta_1 + X_2\beta_2 + \zeta, \tag{10}$$

where $\zeta = D\nu + u$ is the vector of one-way errors with typical element $\zeta_{it} = \nu_i + u_{it}$. This model is another special case of the model with nonspherical errors discussed in the previous section. For example, under the basic setup with $\rho = 0$, we have $\mathrm{cov}(\zeta_{it}, \zeta_{js}) = 0$ for every $i \neq j$ and

$$\mathrm{cov}(\zeta_{it}, \zeta_{is}) = \begin{cases} \sigma_\nu^2 + \sigma_\epsilon^2 & \text{if } t = s, \\ \sigma_\nu^2 & \text{if } t \neq s. \end{cases} \tag{11}$$

Thus, to obtain consistent WALS estimates of β_1 and β_2, we must take into account the stable equicorrelation $\sigma_\nu^2/(\sigma_\nu^2 + \sigma_\epsilon^2)$ exhibited by the one-way errors of the same unit over time.

Section 5 introduces the new `xtwals` command, which is designed for WALS estimation of fixed-effects and random-effects panel-data models. Our fixed-effects and random-effects WALS estimators (labeled as FE-WALS and RE-WALS, respectively) can accommodate both the basic setup with i.i.d. errors and the extended setup with stationary AR(1) errors.

Let us discuss first the basic setup with i.i.d. errors. The FE-WALS estimator exploits the fact that our model-averaging approach satisfies the Frisch–Waugh–Lovell theorem when partialling out an arbitrary subset of focus regressors from the unrestricted model, as shown by De Luca and Magnus (2025).[3] Specializing this result to the subset of focus regressors in the block-diagonal matrix D, this means that the individual effects can be wiped out from model (9) using the familiar within-transformation $v_{it}^* = v_{it} - \bar{v}_i + \bar{v}$ of a variable v_{it}, where $\bar{v}_i = T^{-1}\sum_t v_{it}$ and $\bar{v} = (NT)^{-1}\sum_i\sum_t v_{it}$. Applying this transformation to all variables in (y, X_1, X_2), we first estimate β_1 and β_2 by fitting a WALS regression of y^* on X_1^* and X_2^* with the error variance set equal to the residual sum of squares from the unrestricted model divided by $N(T-1) - k + 1$. This correction for the degrees of freedom in the estimation of σ_ϵ^2 is commonly used to rescale ex-post the variance matrix of the classical fixed-effects estimator (see, e.g., Baltagi 2021, Section 2.2). In WALS, this adjustment serves to rescale the variance matrix, but it also has a direct effect on the model-averaging estimates of β_1 and β_2, because the

3. This result does not extend to the auxiliary regressors, whose coefficients are subject to a nonlinear shrinkage step that invalidates the Frisch–Waugh–Lovell theorem.

unbiased estimator of σ_ϵ^2 affects the t-ratios in the unrestricted model. Moreover, since the WALS estimates depend nonlinearly on the t-ratios of the (transformed) auxiliary coefficients, the estimate of σ_ϵ^2 must be specified at the outset of the estimation procedure. Given the WALS estimates $\tilde{\beta} = (\tilde{\beta}_1', \tilde{\beta}_2')'$ of $\beta = (\beta_1', \beta_2')'$, we can then estimate the ν_i by $\tilde{\nu}_i = \bar{y}_i - \bar{x}_{i,1}'\tilde{\beta}_1 - \bar{x}_{i,2}'\tilde{\beta}_2$, where $\bar{x}_{i,1} = T^{-1}\sum_t x_{it,1}$ and $\bar{x}_{i,2} = T^{-1}\sum_t x_{it,2}$. The estimates resulting from this procedure coincide numerically with those obtained from the more computationally demanding WALS regression of y on $\tilde{X}_1 = (X_1, D)$ and X_2 in model (9).

The RE-WALS estimator of β in the variance component model (10) is based on the FGLS strategy described in Section 2.1. As shown in many textbooks (see, e.g., Baltagi 2021, Section 2.3), the GLS transformation of a variable v_{it} for the case of i.i.d. errors becomes $v_{it}^* = v_{it} - \alpha\bar{v}_i$ with $\bar{v}_i = T^{-1}\sum_t v_{it}$ and

$$\alpha = 1 - \sqrt{\frac{\sigma_\epsilon^2}{T\sigma_\nu^2 + \sigma_\epsilon^2}}.$$

If the panel is unbalanced, the above transformation involves a unit-specific coefficient α_i that is obtained by replacing T with T_i. In the first step of the procedure, we estimate the variance σ_ν^2 of the random effects and the variance σ_ϵ^2 of the idiosyncratic errors by applying the `xtreg,re` command to the unrestricted model (see [XT] **xtreg**). In the second step, we estimate β_1 and β_2 by performing a WALS regression based on the FGLS transformations of the outcome variable and the regressors.

Fixed-effects and random-effects FGLS strategies for the case of AR(1) errors were developed by Bhargava et al. (1982), Baltagi and Li (1991), and Baltagi and Wu (1999). In particular, the random-effects estimator of Baltagi and Wu (1999) extends the estimator of Baltagi and Li (1991) to exogenously unbalanced panels with unequally spaced observations over time. The Stata's `xtregar,fe` command uses the Baltagi–Wu AR(1) transformation to extend this generalization to the fixed-effects estimator. Details on the underlying data transformations can be found in Baltagi and Wu (1999) and the *Methods and formulas* section of [XT] **xtregar**. The FE-WALS algorithm for the extended setup consists of the following steps:

1. estimate ρ and σ_ϵ^2 by running `xtregar,fe` on the unrestricted model;
2. use the estimated correlation $\hat{\rho}$ from step 1 to compute the Baltagi–Wu AR(1) transformations (y^*, X_1^*, X_2^*) of (y, X_1, X_2);
3. compute the within-transformations $(y^{**}, X_1^{**}, X_2^{**})$ of (y^*, X_1^*, X_2^*) by excluding the N observations corresponding to the first period of each unit;
4. estimate β_1 and β_2 by performing a WALS regression of y^{**} on X_1^{**} and X_2^{**} with error variance set equal to its unbiased estimate $\hat{\sigma}_\epsilon^2$ from step 1.

Similarly, the key steps of the RE-WALS algorithm are:

1. estimate ρ, σ_ϵ^2, and σ_ν^2 by running `xtregar,re` on the unrestricted model;

2. use the estimated correlation $\hat{\rho}$ from step 1 to compute the Baltagi–Wu AR(1) transformations (y^*, X_1^*, X_2^*) of (y, X_1, X_2);

3. use the estimated variance components $\hat{\sigma}_\epsilon^2$ and $\hat{\sigma}_\nu^2$ from step 1 to compute the Baltagi–Wu FGLS transformations $(y^{**}, X_1^{**}, X_2^{**})$ of (y^*, X_1^*, X_2^*);

4. estimate β_1 and β_2 by performing a WALS regression of y^{**} on X_1^{**} and X_2^{**}.

We offer two further remarks on these algorithms. First, `xtregar` offers seven alternative estimators of ρ, including the one-step estimator of Baltagi and Wu (1999). Although the FE-WALS and RE-WALS estimators employ the same estimate of ρ, their first-step estimates of σ_ϵ^2 are different (see [XT] **xtregar**). Second, for the within-transformations in the third step of the FE-WALS algorithm to wipe out the individual effects, we must exclude the first observation of each unit. Hence, in addition to modelling only time-varying regressors, this estimator requires at least three observations on each unit. On the other hand, given that the AR(1) transformations ensure homoskedasticity (see Baltagi and Wu 1999, p. 816), the FGLS transformations in the third step of the RE-WALS algorithm allow us to account for the equicorrelation due to the random effects by preserving the observations of the first time period.

2.3 Generalized linear models

Our GLM setup for cross-sectional data assumes that the elements y_i of $y = (y_1, \ldots, y_n)'$ are realizations of independently distributed random variables with mean μ_i, finite nonzero variance σ_i^2, and a distribution which belongs to the one-parameter linear exponential family (LEF) with density or probability mass function

$$f(y_i; \xi_i) = \exp\left[\, y_i \xi_i - b(\xi_i) + c(y_i) \,\right], \tag{12}$$

where ξ_i is the canonical parameter, $b(\cdot)$ is a known, strictly convex and twice continuously differentiable function, and $c(\cdot)$ is a known function. Different choices of $b(\cdot)$ and $c(\cdot)$ result in different distributions within the LEF, such as the normal, gamma, binomial, and Poisson distributions. In the original formulation of Nelder and Wedderburn (1972), the density of y_i also includes a dispersion parameter which, without loss of generality, we set equal to one. The mean and variance of y_i are given by $\mu_i = \mu(\xi_i)$ and $\sigma_i^2 = \sigma^2(\xi_i)$, with $\mu(\xi) = db(\xi)/d\xi$ and $\sigma^2(\xi) = d^2b(\xi)/d\xi^2 = d\mu(\xi)/d\xi$ (see, e.g., McCullagh and Nelder 1989, p. 29).

To model the mean of the outcome in terms of the focus and auxiliary regressors, we assume that there exist a linear predictor

$$\eta_i = \eta_i(\beta) = x_i'\beta = x_{i1}'\beta_1 + x_{i2}'\beta_2, \tag{13}$$

and an invertible and differentiable link function $g(\cdot)$ such that $\eta_i = \eta_i(\beta) = g(\mu(\xi_i)) = g(\mu_i)$ or, equivalently,

$$\mu_i = \mu(\xi_i) = h(\eta_i(\beta)) = h(\eta_i), \tag{14}$$

where $h(\cdot) = g^{-1}(\cdot)$ is the inverse link. This implies that $\xi_i = \xi(\eta_i)$ is a smooth function of η_i with $\xi(\cdot) = \mu^{-1}(h(\cdot))$. When $h(\cdot) = \mu(\cdot)$, the link $g(\cdot)$ is called *canonical*, as it leads to a linear model $\xi_i = \eta_i$ for the canonical parameter.

As usual, our objective is to consider the space of 2^{k_2} models that include all focus regressors and arbitrary subsets of the auxiliary regressors. The jth model is a GLM of the form (12)–(14) under the restriction $S_j'\beta_2 = 0$, where S_j is a $k_2 \times r_j$ selection matrix of rank $0 \le r_j \le k_2$ such that $S_j' = (I_{r_j}, 0)$ (or a column permutation thereof) specifies the auxiliary regressors excluded from the jth model and r_j specifies the number of excluded variables. The loglikelihood for the unrestricted model is

$$\ell(\beta) = c + \sum_{i=1}^{n} \left[\xi_i\, y_i - b(\xi_i)\right], \tag{15}$$

where c is a constant and $\xi_i = \xi(\eta_i)$ depends on β through (14). The ML estimator of β in the jth model maximizes $\ell(\beta)$ subject to $S_j'\beta_2 = 0$ or, equivalently, solves the system of $k_1 + k_2 + r_j$ equations

$$0 = s_1(\beta), \qquad 0 = s_2(\beta) - S_j\nu_j, \qquad 0 = S_j'\beta_2, \tag{16}$$

where ν_j now denotes the $r_j \times 1$ vector of Lagrange multipliers associated with the constraint $S_j'\beta_2 = 0$, and $s_1(\beta)$ and $s_2(\beta)$ are subvectors of the score:

$$s_p(\beta) = \frac{\partial \ell(\beta)}{\partial \beta_p} = \sum_{i=1}^{n} v_i(\beta)\left[y_i - \mu_i(\beta)\right] x_{ip} \qquad (p = 1, 2),$$

with $v_i = d\xi/d\eta_i$. One issue in extending the WALS approach to GLMs is that, except for the normal distribution, the system of likelihood equations (16) is nonlinear and has to be solved by some iterative scheme such as the Newton–Raphson or Fisher scoring methods. De Luca et al. (2018) suggested using the one-step ML estimators, that is, the estimators obtained in the first step of these iterative methods. Expanding the likelihood equations (16) around a starting value $\bar\beta = (\bar\beta_1', \bar\beta_2')'$, the Newton–Raphson method yields the approximations

$$\begin{aligned} 0 &= \bar s_1 + \bar H_{11}(\beta_1 - \bar\beta_1) + \bar H_{12}(\beta_2 - \bar\beta_2), \\ 0 &= \bar s_2 + \bar H_{21}(\beta_1 - \bar\beta_1) + \bar H_{22}(\beta_2 - \bar\beta_2) - S_j\nu_j, \\ 0 &= S_j'\beta_2, \end{aligned} \tag{17}$$

where $\bar s_p = s_p(\bar\beta)$ and $\bar H_{pq} = H_{pq}(\bar\beta)$ denotes the (p,q)-block of the Hessian matrix evaluated at $\bar\beta$, with

$$H_{pq}(\beta) = \frac{\partial^2 \ell(\beta)}{\partial \beta_p \partial \beta_q'} = \sum_{i=1}^{n} \psi_i(\beta) x_{ip} x_{iq}' \qquad (p, q = 1, 2),$$

$\psi_i = \omega_i(y_i - \mu_i) - v_i^2\,\sigma_i^2$, and $\omega_i = d^2\xi/d\eta_i^2$. Similarly, the Fisher scoring method gives

$$\begin{aligned} 0 &= \bar s_1 - \bar{\mathcal{F}}_{11}(\beta_1 - \bar\beta_1) - \bar{\mathcal{F}}_{12}(\beta_2 - \bar\beta_2), \\ 0 &= \bar s_2 - \bar{\mathcal{F}}_{21}(\beta_1 - \bar\beta_1) - \bar{\mathcal{F}}_{22}(\beta_2 - \bar\beta_2) - S_j\nu_j, \\ 0 &= S_j'\beta_2, \end{aligned} \tag{18}$$

where $\bar{\mathcal{F}}_{pq} = \mathcal{F}_{pq}(\bar{\beta})$ denotes the (p,q)-block of the expected information matrix evaluated at $\bar{\beta}$, with

$$\mathcal{F}_{pq}(\beta) = -\,\mathbb{E}\left(H_{pq}(\beta)\right) = \sum_{i=1}^{n} v_i^2(\beta)\,\sigma_i^2(\beta)\,x_{ip}x_{iq}' \qquad (p,q=1,2).$$

In the canonical link case, the above expressions simplify considerably as $\xi_i = \eta_i$, $v_i = 1$, $\omega_i = 0$, and $\psi_i = -\sigma_i^2$ for all observations, and the Newton–Raphson and Fisher scoring methods coincide because $\mathcal{F}_{pq}(\beta) = -H_{pq}(\beta)$ for every β.

The one-step ML estimators of β that solve either (17) or (18) are asymptotically equivalent to the fully-iterated ML estimators and admit closed-form expressions. Consider, for example, the linearized likelihood equations in (18) for the unrestricted model with $S_j = 0$. In matrix form, Fisher scoring relies on the data transformations

$$\bar{y} = \bar{X}_1\bar{\beta}_1 + \bar{X}_2\bar{\beta}_2 + \bar{u}, \qquad \bar{X}_1 = \bar{\Sigma}^{1/2}\bar{V}X_1, \qquad \bar{X}_2 = \bar{\Sigma}^{1/2}\bar{V}X_2, \tag{19}$$

where X_1 and X_2 are the matrices of focus and auxiliary regressors, $\bar{u} = \bar{\Sigma}^{-1/2}(y-\bar{\mu})$, $\bar{\Sigma} = \mathrm{diag}(\sigma_1^2(\bar{\beta}),\ldots,\sigma_n^2(\bar{\beta}))$, $\bar{V} = \mathrm{diag}(v_1(\bar{\beta}),\ldots,v_n(\bar{\beta}))$, and $\bar{\mu} = (\mu_1(\bar{\beta}),\ldots,\mu_n(\bar{\beta}))'$. Since $\bar{X}_p'\bar{u} = \bar{s}_p$ and $\bar{X}_p'\bar{X}_q = \bar{\mathcal{F}}_{pq}$ $(p,q=1,2)$, the unrestricted one-step ML estimator of β can be written in closed form as

$$\hat{\beta}_{1u} = (\bar{X}_1'\bar{X}_1)^{-1}\bar{X}_1'\bar{y} - (\bar{X}_1'\bar{X}_1)^{-1}\bar{X}_1'\bar{X}_2\hat{\beta}_{2u}, \qquad \hat{\beta}_{2u} = (\bar{X}_2'\bar{M}_1\bar{X}_2)^{-1}\bar{X}_2'\bar{M}_1\bar{y}, \tag{20}$$

where $\bar{M}_1 = I_n - \bar{X}_1(\bar{X}_1'\bar{X}_1)^{-1}\bar{X}_1'$. These are the LS coefficients in the linear regression of $\bar{y}$ on $\bar{X}_1$ and $\bar{X}_2$. Rewriting $\bar{y}$ in (19) as

$$\bar{y} = \bar{\Sigma}^{1/2}\bar{V}(X_1\bar{\beta}_1 + X_2\bar{\beta}_2 + \bar{u}^*) = \bar{\Sigma}^{1/2}\bar{V}\bar{y}^*, \tag{21}$$

with $\bar{u}^* = \bar{\Sigma}^{-1}\bar{V}^{-1}(y-\bar{\mu})$, $\hat{\beta}_{1u}$ and $\hat{\beta}_{2u}$ can also be interpreted as the LS coefficients in the weighted regression of $\bar{y}^*$ on X_1 and X_2 with weights equal to the diagonal elements of $\bar{\Sigma}\bar{V}^2$ (i.e., the first step of the IRLS procedure). These results extend to the one-step ML estimators of β for the jth model (De Luca et al. 2018, Proposition 1) and they provide the data transformations needed to operationalize the WALS approach in the class of GLMs.

Section 6 introduces the new `glmwals` command for WALS estimation of univariate GLMs. There are two variants of the WALS estimates for GLMs. The one-step estimates are obtained by fitting a weighted WALS regression of $\bar{y}^*$ on X_1 and X_2 with (rescaled) importance weights equal to the diagonal elements of $\bar{\Sigma}\bar{V}^2$, while the iterative estimates are based on an iterative procedure that repeatedly updates the starting value $\bar{\beta}$ using the one-step WALS estimates from the previous iteration until convergence.

3 The `hetwals` command

The command `hetwals` provides WALS estimates of a linear model with multiplicative heteroskedasticity. Its syntax is as follows:

`hetwals` *depvar* [`(`*focvars*`)`] *auxvars* [*weight*] [*if*] [*in*] `, het(`*hetvars*`)`
[*wals_options het_options*]

where *depvar*, `(`*focvars*`)`, *auxvars*, and *wals_options* are defined as in the `wals` command (version 3.0), the `het(`*hetvars*`)` option specifies the regressors of the variance function, and *het_options* denote the additional options listed in Table 1.[4] Notice that, even if *hetvars* always includes a constant term, the `het` option cannot be omitted as this would result in a homoskedastic linear model that can be estimated by the `wals` command. The regressors to be specified in *hetvars* are usually, though not necessarily, related to those in `(`*focvars*`)` and *auxvars*. In particular, the two sets of regressors for the mean and variance functions may also coincide.

Table 1: Additional options of the `hetwals` command

het_options	Description
Estimating the variance function	
`twostep`	use the two-step GLS estimator; default is the ML estimator
`hconstraints(numlist|matname)`	specifies linear constraints for the variance function
maximize_options	control features of the maximization procedure
Reporting	
`waldhet`	perform the Wald test for homoskedasticity instead of the likelihood ratio test
`showhetreg`	show first-step estimates of unrestricted model

3.1 Options

Estimating the variance function. In the first step, `hetwals` estimates the variance function from the unrestricted model using one of the two methods available in the `hetregress` command: the (default) ML estimator or Harvey's two-step GLS estimator. We offer the following options:

`twostep` specifies the two-step GLS estimator, instead of the default ML estimator. This estimation method does not support weights.

`hconstraints(numlist|matname)` specifies linear constraints to be applied in the first-step estimates of the variance function (see [R] **constraint** and [P] **makecns**). This option can only be used with the default ML estimator.

If the ML estimator for the unrestricted model does not converge, then `hetwals` aborts with an error. When this happens, one can still control the default features of the

4. For details on the *wals_options* see De Luca and Magnus (2024, Section 3). The `hetwals` command does not support the `sigma(#)` option.

maximization process using the *maximize_options* (see [R] **Maximize**):

`tech`nique(*algorithm_spec*), `tol`erance(#), `ltol`erance(#), `nrtol`erance(#), `nonrtol`erance(#), `iter`ate(#), `from`(*init_spec*), and `dif`ficult.

In the second step, `hetwals` estimates the mean function by a weighted WALS regression of *depvar* on (*focvars*) and *auxvars* with analytic weights equal to the reciprocals of the estimated variances.

Reporting. The output's header of `hetwals` displays a diagnostic test for homoskedasticity in the unrestricted model. In contrast, the coefficient table is restricted to the second-step WALS estimates of the focus and auxiliary parameters in the mean function. In addition to the `noheader`, `nofocus`, `noauxiliary`, `notable`, and `cformat(%fmt)` options of the `wals` command, `hetwals` supports the following reporting options:

`waldhet` displays the Wald test for homoskedasticity in the unrestricted model instead of the likelihood ratio test.

`showhetreg` displays the first-step estimates of the unrestricted model.[5]

The first-step estimates of the parameters in the variance function and their variance matrix are stored in `e(hb)` and `e(hV)`, respectively. A complete list of the results stored by `hetwals` is presented in Appendix A.1.

3.2 Predictions

The `predict` command for `hetwals` has two possible syntaxes. The first is

`predict` [*type*] *newvar* [*if*] [*in*] , [`xb biasp stdp rmsep bcp stdbcp residuals bcresiduals sigma`]

where `xb`, the default, calculates the linear prediction (which is the WALS predictor); `biasp`, `stdp` and `rmsep` calculate, respectively, the plug-in estimates of the bias, SE and RMSE of the linear prediction; `bcp` calculates the bias-corrected linear prediction; `stdbcp` calculates the SE of the bias-corrected linear prediction; `residuals` calculates the residuals from the linear prediction; `bcresiduals` calculates the residuals from the bias-corrected linear prediction; and `sigma` calculates the estimated standard deviations $\hat{\sigma}_i = \exp(\hat{\alpha}' v_i/2)$ of the errors. The first eight statistics are defined as in the `predict` command for `wals` (see De Luca and Magnus 2024, Section 5.1) and they are all conditional on the first-step estimates of the variance function. The only statistic allowed with the `margwals` command is the default linear prediction.

The second syntax of the `predict` command for `hetwals` is

`predict` [*type*] *newvar*$_l$ *newvar*$_u$ [*if*] [*in*] , [`cinterval pinterval level(#)`

5. For the ML estimator, this option can also be specified together with the `nolog` option to suppress the display of the iteration log.

`rseed(#)`]

where `cinterval` calculates the confidence interval for $\mathbb{E}(y_i|x_i)$, `pinterval` calculates the prediction interval for $y_i|(x_i, v_i)$, `level` sets the confidence level, and `rseed` sets the random-number seed to ensure reproducibility of the prediction interval for $y_i|(x_i, v_i)$. The lower and upper bounds of these intervals are stored in the new variables $newvar_l$ and $newvar_u$. The confidence interval $\mathbb{E}(y_i|x_i)$ is based on the empirical percentiles of $\tilde{B}^* x_i$, where $\tilde{B}^*$ is the $R \times k$ matrix of MC replications of the bias-corrected WALS estimator $\tilde{\beta}^*$ of β stored in `e(bcsimdata)`. In contrast, the prediction interval for $y_i|(x_i, v_i)$ is based on the empirical percentiles of $\tilde{B}^* x_i + \hat{\sigma}_i \epsilon$, where $\hat{\sigma}_i = \exp(\hat{\alpha}' v_i/2)$ and ϵ is an $R \times 1$ vector of $\mathrm{N}(0, 1)$ random draws.

4 The `ar1wals` command

The `ar1wals` command provides WALS estimates of a linear model with stationary AR(1) errors. Its syntax is:

`ar1wals` *depvar* [`(`*focvars*`)`] *auxvars* [*if*] [*in*], [*wals_options* *ar1_options*]

where *depvar*, `(`*focvars*`)`, *auxvars*, and *wals_options* are defined as in the `wals` command (version 3.0), and *ar1_options* denote the additional options listed in Table 2.

Table 2: Additional options of the `ar1wals` command

ar1_options	Description
Estimating the autocorrelation coefficient ρ	
`rho`**`type`**`(`*rhomethod*`)`	specify the estimation method
`two`**`step`**	stop after the first iteration
`sse`**`search`**	specify the Hildreth–Lu estimation procedure
optimize_options	control features of the optimization process
AR(1) transformations	
`corc`	use the Cochrane–Orcutt transformation; default is the Prais–Winsten transformation
Reporting	
`showar1`	show first-step estimates of the unrestricted model

Notice that, before using `ar1wals`, the dataset in memory must be declared as a time-series ([TS] **tsset**). Moreover, `(`*focvars*`)` and *auxvars* cannot include time-series transformations of *depvar*, and weights are not allowed.

4.1 Options

Estimating autocorrelation. In the first step, `ar1wals` estimates the autocorrelation coefficient ρ from the unrestricted model, using one of the six methods supported by the `prais` command. The estimation of ρ is controlled by the following options:

`rho`type(*rhomethod*) specifies the estimation method for ρ, where the available choices for *rhomethod* are:

`regress`	LS estimator in a single-lag residual regression
`freg`	LS estimator in a single-lead residual regression
`tscorr`	sample autocorrelation of residuals
`dw`	autocorrelation based on Durbin–Watson statistic
`theil`	adjusted Theil's sample autocorrelation
`nagar`	adjusted Theil-Nagar's autocorrelation

The default estimation method is `regress`. For a description of the various methods, see Judge et al. (1985, Chapter 8) and [TS] **prais**.

`twostep` specifies that the chosen estimation method for ρ terminates after the first iteration. By default all methods are iterated until convergence, but the underlying estimators are efficient at each step.

`ssesearch` specifies the Hildreth–Lu estimation procedure, which uses a grid search to find the value of ρ that minimizes the sum-of-squared errors in the Cochrane–Orcutt or Prais–Winsten transformation of the unrestricted model (see Hildreth and Lu 1960; Davidson and MacKinnon 1993, p. 333; and [TS] **prais**).

There are two additional options within *optimize_options*: `iterate(#)` sets the maximum number of iterations in estimating ρ, while `tolerance(#)` sets the tolerance for the convergence of the `prais` estimates of the unrestricted model. The default values of these options are `iterate(300)` and `tolerance(1e-6)`.

AR(1) transformations. In the second step, `ar1wals` estimates a WALS regression based on the Cochrane–Orcutt or Prais–Winsten transformations of *depvar*, (*focvars*) and *auxvars*. The Prais–Winsten transformation allows preserving the first observation, which is lost in the Cochrane–Orcutt transformation. The choice between the two transformations is determined by the option:

`corc` specifies that the Cochrane–Orcutt transformation must be used instead of the default Prais–Winsten transformation.

Reporting. The first-step estimate of ρ is always reported in the output's header of `ar1wals`, but one can also specify the `showar1` option to display the `prais` estimates of the unrestricted model.[6]

The first-step estimate of ρ is stored in `e(rho)`. A complete list of the results stored by `ar1wals` is presented in Appendix A.2.

6. The `showar1` option can also be specified together with `nolog` to suppress the display of the iteration log when estimating the unrestricted model.

4.2 Predictions

The `predict` command for `ar1wals` has two possible syntaxes. The first is

`predict` [*type*] *newvar* [*if*] [*in*], [`xb biasp stdp rmsep bcp stdbcp residuals bcresiduals wp bcwp`]

where `xb`, the default, calculates the linear prediction (without accounting for the estimated correlation among residuals); `biasp`, `stdp` and `rmsep` calculate, respectively, the plug-in estimates of the bias, SE and RMSE of the linear prediction; `bcp` calculates the bias-corrected linear prediction; `stdbcp` calculates the SE of the bias-corrected linear prediction; `residuals` calculates the residuals from the linear prediction; `bcresiduals` calculates the residuals from the bias-corrected linear prediction; `wp` calculates the s periods-ahead WALS predictions; and `bcwp` calculates the s periods-ahead bias-corrected WALS predictions. The first eight statistics are defined as in the other WALS commands, while the last two statistics are defined as in Goldberger (1962). Specifically, for any $s \geq 1$, the WALS predictor of y_{T+s} is given by $\tilde{y}_{T+s} = x'_{T+s}\tilde{\beta} + \hat{\rho}^s \tilde{e}_T$, where $\tilde{\beta}$ is the WALS estimate of β, $\hat{\rho}$ is the first-step estimate of ρ, and $\tilde{e}_T = y_T - x'_T\tilde{\beta}$ is the WALS residual in the last estimation period. The bias-corrected WALS predictor of y_{T+s} is defined similarly by replacing $\tilde{\beta}$ and $\tilde{e}_T$ with $\tilde{\beta}^*$ and $\tilde{e}^*_T = y_T - x'_T\tilde{\beta}^*$, respectively. All statistics are conditional on $\hat{\rho}$ and the only statistic allowed with the `margwals` command is the default linear prediction.

The second syntax of the `predict` command for `ar1wals` is

`predict` [*type*] *newvar*$_l$ *newvar*$_u$ [*if*] [*in*], [`cinterval pinterval level(#) rseed(#)`]

where `cinterval` calculates the confidence interval for $\mathbb{E}(y_{T+s}|x_{T+s})$, `pinterval` calculates the prediction interval for $y_{T+s}|x_{T+s}$, `level` sets the confidence level, and `rseed` sets the random-number seed to ensure reproducibility of the prediction interval for $y_{T+s}|x_{T+s}$. The lower and upper bounds of these intervals are stored in the new variables *newvar*$_l$ and *newvar*$_u$. The confidence interval for $\mathbb{E}(y_{T+s}|x_{T+s})$ is based on the empirical percentiles of $\tilde{B}^* x_{T+s} + \hat{\rho}^s \tilde{E}_T$, where $\tilde{B}^*$ is the $R \times k$ matrix of MC replications of $\tilde{\beta}^*$ and $\tilde{E}_T = y_T \iota_R - \tilde{B}^* x_T$. The prediction interval for $y_{T+s}|x_{T+s}$ is instead based on the empirical percentiles of $\tilde{B}^* x_{T+s} + \hat{\rho}^s \tilde{E}_T + \hat{\sigma}_\epsilon \epsilon$, where $\hat{\sigma}_\epsilon$ is the LS estimate of σ_ϵ and ϵ is an $R \times 1$ vector of N$(0,1)$ random draws.

5 The `xtwals` command

The `xtwals` command provides WALS estimates of fixed-effects and random-effects panel-data models with either i.i.d. or AR(1) errors. Its syntax is as follows:

`xtwals` *depvar* [(*focvars*)] *auxvars* [*weight*] [*if*] [*in*], [*wals_options xt_options*]

where *depvar*, (*focvars*), *auxvars*, and *wals_options* are defined as in the `wals` command (version 3.0), and *xt_options* denote the additional options listed in Table 3.[7]

Table 3: Additional options of the `xtwals` command

xt_options	Description
Model setup	
`re`	specify the random-effects approach; the default approach is fixed-effects
`ar1`	specify the extended setup with AR(1) errors; the default setup assumes i.i.d. errors
Estimating variances and autocorrelation	
`sa`	use the Swamy–Arora estimator of the variance components; only for `re` with i.i.d. errors
`rhotype(`*rhomethod*`)`	specify the estimation method for ρ
`rhof(#)`	set ρ equal to a specific value #
`twostep`	compute the two-step estimate of ρ
Reporting	
`showxtreg`	show first-step estimates of the unrestricted model

Before using `xtwals`, the dataset in memory must be declared as a panel dataset. In particular, the setups with i.i.d. errors require declaring at least the panel variable, while the extended setups with AR(1) errors require declaring both the panel and time variables (see [XT] **xtset**, [XT] **xtreg** and [XT] **xtregar**). In either case, (*focvars*) and *auxvars* cannot include lags of *depvar*, because of the strict exogeneity assumption.

5.1 Options

Model setup. The model setup of `xtwals` depends on the following options:

`re` specifies that the random-effects approach must be used instead of the default fixed-effects approach.

`ar1` specifies that the extended setup with AR(1) errors must used instead of the default setup with i.i.d. errors.

The default is the fixed-effects model with i.i.d. errors, which does not require the specification of either of the two options. In contrast, the random-effects model with i.i.d. errors requires the specification of the `re` option; the fixed-effects model with AR(1) errors requires the specification of the `ar1` option; and the random-effects model with AR(1) errors requires the specification of both options. Each model setup implies certain restrictions on other aspects of the model. Specifically, in the fixed-effects models, (*focvars*) and *auxvars* cannot contain time-invariant regressors, the `auxconstant` option

7. The `xtwals` command does not support the `noconstant` and `sigma(#)` options.

is not allowed, and the availability of weights is restricted to analytic or frequency weights that are constant within units of the panel.[8] The random-effects models do not support weights. Finally, in the models with AR(1) errors, (*focvars*) and *auxvars* cannot include the time variable.

Estimating variances and autocorrelation. As discussed in Section 2.2, the data transformations used by `xtwals` may depend on the autocorrelation coefficient ρ and the variance components σ_ν^2 and σ_ϵ^2. These parameters are estimated from the unrestricted model using the `xtreg` command for the basic setups with i.i.d. errors and the `xtregar` command for the extended setups with AR(1) errors. There are four options related to this aspect of the estimation procedure:

`sa` uses the small-sample Swamy–Arora estimator of the variance of the random effects instead of the default consistent estimator. This option can be specified only in the random-effects model with i.i.d. errors.

`rhotype(`*rhomethod*`)` specifies the estimation method for ρ, where the available choices for *rhomethod* are:

`regress`	LS estimator in a single-lag residual regression
`freg`	LS estimator in a single-lead residual regression
`tscorr`	sample autocorrelation of residuals
`dw`	autocorrelation based on Durbin–Watson statistic
`theil`	adjusted Theil's sample autocorrelation
`nagar`	adjusted Theil–Nagar's autocorrelation
`onestep`	Baltagi–Wu's one-step estimator

The default method is `dw`. For details see [XT] **xtregar**.

`twostep` requests that the first six estimation methods for ρ are stopped after the first iteration. By default, these methods are iterated until convergence.

`rhof(#)` requests that ρ is set equal to a value # in the $[-1, 1]$ interval.

Note that `rhotype`, `twostep`, and `rhof(#)` must be specified together with `ar1`; `twostep` cannot be specified together with `rhotype(onestep)`; and `rhof(#)` cannot be specified together with `rhotype` and `twostep`.

Reporting. The output's header of `xtwals` displays the panel variable, the number of panels, and the minimum and maximum number of observations per panel. In the models with AR(1) errors, it also displays the time variable and the estimated value of ρ. The coefficient table is always restricted to the estimates obtained in the last WALS regression on the transformed variables. To display the preliminary `xtreg`/`xtregar` estimates of the unrestricted model one can specify the `showxtreg` option.

The results stored by `xtwals` are listed in Appendix A.3.

8. In the fixed-effects model with AR(1) errors, the only estimation methods for ρ that support weights are `regress` and `freg`.

5.2 Predictions

The `predict` command for `xtwals` has two possible syntaxes. The first is

`predict` [*type*] *newvar* [*if*] [*in*] , [`xb biasp stdp rmsep bcp stdbcp ie bcie residuals bcresiduals wp bcwp`]

where `xb`, the default, calculates the linear prediction (without including the individual effects); `biasp`, `stdp`, and `rmsep` calculate, respectively, the plug-in estimates of the bias, SE, and RMSE of the linear prediction; `bcp` calculates the bias-corrected linear prediction; `stdbcp` calculates the SE of the bias-corrected linear prediction; `ie` calculates the estimates of the individual fixed-effects or random-effects; `bcie` calculates the bias-corrected WALS estimates of the individual effects; `residuals` calculates the residuals from the linear prediction and the individual effects; `bcresiduals` calculates the residuals from the bias-corrected linear prediction and the bias-corrected individual effects; `wp` calculates the s periods-ahead WALS predictions; and `bcwp` calculates the s periods-ahead bias-corrected WALS predictions. The last six statistics are currently restricted to the fixed-effects and random-effects models with i.i.d. errors.[9] Let $\tilde{\beta}$ denote either the FE-WALS or the RE-WALS estimates of β and let $\tilde{\beta}^*$ denote the corresponding bias-corrected estimates. Then, for any $i = 1, \ldots, N$, the individual fixed-effects are estimated by $\tilde{\nu}_i = \bar{y}_i - \bar{x}_i'\tilde{\beta}$, while the individual random-effects are estimated by $\tilde{\nu}_i = \hat{\sigma}_\nu^2(\bar{y}_i - \bar{x}_i'\tilde{\beta})/(\hat{\sigma}_\nu^2 + \hat{\sigma}_\epsilon^2/T_i)$. The bias-corrected estimates $\tilde{\nu}_i^*$ are calculated similarly by replacing $\tilde{\beta}$ with $\tilde{\beta}^*$. For any $i = 1, \ldots, N$ and any $t = 1, \ldots, T_i$, the residuals and the bias-corrected residuals are defined as $y_{it} - x_{it}'\tilde{\beta} - \tilde{\nu}_i$ and $y_{it} - x_{it}'\tilde{\beta}^* - \tilde{\nu}_i^*$, respectively. For any $i = 1, \ldots, N$ and any $s \geq 1$, the s periods-ahead WALS prediction of $y_{i,T+s}$ is given by $\tilde{y}_{i,T+s} = x_{i,T+s}'\tilde{\beta} + \tilde{\nu}_i$ (see, e.g., Section 2.5 in Baltagi 2021 and the additional references therein), while the s periods-ahead bias-corrected WALS prediction is given by $\tilde{y}_{i,T+s}^* = x_{i,T+s}'\tilde{\beta}^* + \tilde{\nu}_i^*$. The only statistic allowed with the `margwals` command is the default linear prediction.

The second syntax of the `predict` command for `xtwals` is

`predict` [*type*] *newvar*$_l$ *newvar*$_u$ [*if*] [*in*] , [`cinterval pinterval level(#) rseed(#)`]

where `cinterval` calculates the confidence interval for $\mathbb{E}(y_{i,T+s}|x_{i,T+s})$, `pinterval` calculates the prediction interval for $y_{i,T+s}|x_{i,T+s}$, `level` sets the confidence level, and `rseed` sets the random-number seed to ensure reproducibility of the prediction interval for $y_{i,T+s}|x_{i,T+s}$. As with the s periods-ahead WALS predictions, these intervals are available only for the fixed-effects and random-effects models with i.i.d. errors. The confidence interval for $\mathbb{E}(y_{i,T+s}|x_{i,T+s})$ is based on the empirical percentiles of

9. The documentation of the `predict` command for `xtregar` does not provide detailed information on the estimation of the individual effects. The proposed estimation method seems to be based on Bhargava et al. (1982, equation 16b), but this approach is only valid for the fixed-effects model with equally-spaced observations.

$\tilde{B}^* x_{i,T+s} + \tilde{V}_i^*$, where $\tilde{B}^*$ is the $R \times k$ matrix of MC replications of $\tilde{\beta}^*$ and $\tilde{V}_i^*$ is defined as $\bar{y}_i \imath_R - \tilde{B}^* \bar{x}_i$ in the fixed-effects model and as $\hat{\sigma}_\nu^2 (\bar{y}_i \imath_R - \tilde{B}^* \bar{x}_i)/(\hat{\sigma}_\nu^2 + \hat{\sigma}_\epsilon^2/T_i)$ in the random-effects model. In contrast, the prediction interval for $y_{i,T+s}|x_{i,T+s}$ is based on the empirical percentiles of $\tilde{B}^* x_{T+s} + \tilde{V}_i^* + \hat{\sigma}_\epsilon \epsilon$, where $\hat{\sigma}_\epsilon$ is the LS estimate of σ_ϵ and ϵ is an $R \times 1$ vector of $\mathrm{N}(0,1)$ random draws.

6 The `glmwals` command

The `glmwals` command provides the one-step and iterative WALS estimates of univariate GLMs. Its syntax is as follows:

`glmwals` *depvar* [(*focvars*)] *auxvars* [*if*] [*in*] [*weight*], [*wals_options* *glm_options*]

where *depvar*, (*focvars*), *auxvars*, and *wals_options* are defined as in the `wals` command (version 3.0), and *glm_options* are the additional options listed in Table 4.

Table 4: Additional options of the `glmwals` command

glm_options	Description
GLM setup	
`family(`*familyname*`)`	specify the distribution of *depvar*; default is the `Gaussian` distribution
`link(`*linkname*`)`	specify the link function; default is the canonical link for the specified `family()`
`offset(`*ovar*`)`	include *ovar* in the linear predictor with coefficient constrained to 1
`exposure(`*evar*`)`	include ln(*evar*) in the linear predictor with coefficient constrained to 1
GLM estimation	
`ltolerance(#)`	set tolerance for the IRLS estimates
`iter0(#)`	set maximum iterations for the IRLS estimates
`init(`*imethodname*`)`	set initial values for the mean of *depvar*
`onestep`	specify the one-step WALS estimates
`tolerance(#)`	set tolerance for the iterative WALS estimates
`iterate(#)`	set maximum iterations for the iterative WALS estimates
Reporting	
`showirls`	display the IRLS estimates
`nolog`	suppress iteration log
`eform`	display exponentiated estimates

6.1 Options

GLM setup. In addition to the model setup options of the `wals` command, `glmwals` supports the following options:

`family(`*familyname*`)` specifies the distribution of *depvar*. The available choices for *familyname* are

`gaussian`	`normal`	`igaussian`	`inormal`
`gamma`	`binomial`	`bernoulli`	`poisson`

where `gaussian` is the default distribution, `igaussian` denotes the inverse Gaussian distribution, `normal` is a synonym for `gaussian`, and `inormal` is a synonym for `igaussian`. The binomial distribution can be specified in three ways:

`binomial` `binomial` $\#_N$ `binomial` *varname*$_N$

where `binomial 1` is the same as `binomial` or `bernoulli`, `binomial` $\#_N$ defines an integer scalar $\#_N$ for the number of trials, and `binomial` *varname*$_N$ allows the number of trials to vary across observations according to the values of *varname*$_N$. Distribution names are not case sensitive.

`link(`*linkname*`)` specifies the link function $g(\mu_i) = \eta_i$ that maps the mean outcome μ_i into the linear predictor $\eta_i = x_i'\beta$. The available choices for *linkname* and the associated link functions are

`identity`	$\eta_i = \mu_i$
`log`	$\eta_i = \ln(\mu_i)$
`logit`	$\eta_i = \ln(\mu_i/(1-\mu_i))$
`probit`	$\eta_i = \Phi^{-1}(\mu_i)$
`cloglog`	$\eta_i = \ln(-\ln(1-\mu_i))$
`loglog`	$\eta_i = -\ln(-\ln(\mu_i))$
`logc`	$\eta_i = \ln(1-\mu_i)$
`reciprocal`	$\eta_i = \mu_i^{-1}$
`power` a	$\eta_i = \mu_i^a$
`opower` a	$\eta_i = (\mu_i^a/(1-\mu_i)^a - 1)/a$

where $\Phi(\cdot)$ is the standard-normal distribution function and a is a real scalar. The default link is the canonical link of each family, that is:

`family(gaussian)`	`link(identity)`
`family(igaussian)`	`link(power -2)`
`family(binomial)`	`link(logit)`
`family(poisson)`	`link(log)`
`family(gamma)`	`link(power -1)`

`offset(`*ovar*`)` specifies that the *ovar* variable must be included in the linear predictor with coefficient constrained to be 1.

`exposure(`*evar*`)` specifies that the logarithm of the *evar* variable must be included in the linear predictor with coefficient constrained to be 1.

Except for `family(nbinomial)` and `link(nbinomial)`, `glmwals` supports all combinations of *familyname* and *linkname* allowed in the `glm` command (see [R] **glm**). Note, however, that our command aborts with an error when nonstandard combinations of *familyname* and *linkname* lead to convergence problems in the preliminary IRLS estimates of the unrestricted model. By default, the scale parameter for the WALS regressions is set equal to 1 for discrete distributions (i.e., `binomial` and `poisson`) and equal to the Pearson χ^2 statistic divided by the residual degrees of freedom for continuous distributions (i.e., `gaussian`, `igaussian`, and `gamma`). To specify alternative values of this parameter one can use the `sigma(#)` option.[10]

GLM estimation. `glmwals` also supports the following estimation options:

`ltolerance(#)` specifies the tolerance for the deviance in the IRLS estimates of the unrestricted model. The default is `ltolerance(1e-6)`.

`iter0(#)` specifies the maximum number of iterations for the IRLS estimates of the unrestricted model. The default is `iter0(300)`.

`init(`*imethodname*`)` specifies the initial values for μ_i needed to initialize the WALS estimation procedure. We offer four choices for *imethodname*: `irls_u` (the default) uses the predicted values obtained from the unrestricted model; `irls_r` uses the predicted values obtained from the fully restricted model; while `irls_0` and `irls_0` *varname* reproduce, respectively, the default initial values and the `mu(`*varname*`)` option of the `glm` command.

`onestep` requests the one-step WALS estimates instead of the default iterative WALS estimates.

`tolerance(#)` specifies the tolerance for the convergence criterion of the iterative WALS estimates, which is the relative difference in the vector of estimated coefficients from one iteration to the next. The iterative procedure is initialized using the one-step WALS estimates, and convergence is declared when the criterion is smaller than the specified tolerance level. The default is `tolerance(1e-6)`.

`iterate(#)` specifies the maximum number of iterations for the iterative WALS estimates. The default is `iterate(300)`. If convergence is not achieved before reaching the specified threshold, then `glmwals` displays the current results and then aborts with an error.

Reporting. `glmwals` also supports three additional reporting options:

`showirls` displays the IRLS estimates of the unrestricted model.

`nolog` suppresses the display of iteration log in the iterative WALS estimates. If specified together with the `showirls`, this option also suppresses the display of iteration log in the IRLS estimates of the unrestricted model.

`eform` displays the exponentiated estimates $e^{\tilde{\beta}_h}$. This option only affects the displayed

10. For continuous distributions, another common choice is to set the scale parameter equal to the deviance divided by the residual degrees of freedom (e.g., the `scale(dev)` option of `glm`).

results and it can be specified during estimation or on replay. In this case, the plug-in estimates of the bias, SE, and RMSE are multiplied by $e^{\tilde{\beta}_h}$, while the `eform` confidence intervals are reported with exponentiated endpoints (see [R] **nlcom**).

The results stored by `glmwals` are listed in Appendix A.4.

6.2 Predictions

The syntax of the `predict` command for `glmwals` is

`predict` [*type*] *newvar* [*if*] [*in*] , [`mu xb biasp stdp rmsep bcp stdbcp residuals nooffset`]

where `mu`, the default, calculates the plug-in estimate $\tilde{\mu}_i = h(x_i'\tilde{\beta})$ of μ_i; `xb` calculates the estimate $\tilde{\eta}_i = x_i'\tilde{\beta}$ of η_i; `biasp`, `stdp`, and `rmsep` calculate, respectively, the plug-in estimates of the bias, SE, and RMSE of $\tilde{\eta}_i$; `bcp` calculates the bias-corrected WALS estimate $\tilde{\eta}_i^* = x_i'\tilde{\beta}^*$ of η_i; `stdbcp` calculates the SE of $\tilde{\eta}_i^*$; `residuals` calculates the so-called response residuals $y_i - \tilde{\mu}_i$; and `nooffset` modifies the default calculations made by `predict` to ignore the offset variable (if applicable). As discussed in De Luca and Magnus (2024, Section 5.3), we do not compute the estimated bias of $\tilde{\mu}_i$ as an estimator of μ_i. However, the `margwals` command can be applied to the default `mu` option (and the `xb` option, if desired) to compute point estimates and simulation-based confidence intervals for mean responses and marginal effects at given values of the regressors.

7 The Hong Kong housing market

In this section, we present the `hetwals` estimates of a heteroskedastic hedonic housing price model for the Hong Kong residential property market, using data from Magnus et al. (2011). The data cover $n = 560$ transactions over the period 2004–2007 at the housing estate 'South Horizon,' which is one of the most densely populated private-housing estates in the Southern District of Hong Kong, with a total of 9812 apartments in 34 blocks, each with 25–42 floors. Our dependent variable is the natural logarithm of the sales price per square foot (`LPRICE`), and the regressors include various features of the apartments as indicated in the following description:

```
. use "MWZ_2011", clear
. describe
Contains data from Data/MWZ_2011.dta
 Observations:           560
    Variables:            13                  16 Jun 2024 19:17
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
LPRICE          double  %10.0g               ln sales price per square foot
LAREA           double  %10.0g               ln size in square feet
LFLOOR          double  %10.0g               ln floor level
```

```
GARV            byte    %10.0g                1 if garden view; 0 otherwise
INDV            byte    %10.0g                1 if industry view; 0 otherwise
SEAVF           byte    %10.0g                1 if full sea view; 0 otherwise
SEAVS           byte    %10.0g                1 if semi-sea view; 0 otherwise
SEAVM           byte    %10.0g                1 if minor sea view; 0 otherwise
MONV            byte    %10.0g                1 if mountain view; 0 otherwise
STRI            byte    %10.0g                1 if internal street view; 0
                                                otherwise
STRN            byte    %10.0g                1 if no street view; 0 otherwise
UNLUCK          byte    %10.0g                1 if located on floors 4,14,24,34;
                                                0 otherwise
BLOCK           byte    %10.0g                Block number
```

Detailed information about the regressors can be found in Magnus et al. (2011).[11] The analysis aims to investigate the contribution of the various features to an apartment's sales price per square foot within the South Horizon housing estate. Based on a preliminary analysis of the data, the authors assumed a linear model with multiplicative heteroskedasticity of the form $\sigma_i^2 = \exp(\alpha_2 \mathrm{STRN}_i)$ and $\alpha_1 = 0$. Below, we first reproduce their ML estimates and their WALS estimates based on a Laplace prior:

```
. local y                "LPRICE"
. local X1               "LAREA LFLOOR GARV INDV SEAVF"
. local X2               "SEAVS SEAVM MONV STRI STRN UNLUCK"
. local V                "STRN"
. local X                `X1´ `X2´
. local hetregr_ops      "nolog waldhet"
. local hetwals_ops      "rseed(123) prior(Laplace) waldhet"
. constraint 1 [lnsigma2]_cons = 0
. hetregress `y´ `X´, het(`V´) constraints(1) `hetregr_ops´
Heteroskedastic linear regression               Number of obs     =        560
ML estimation
                                                Wald chi2(11)     =     432.64
Log likelihood =  61.89692                      Prob > chi2       =     0.0000
 ( 1)  [lnsigma2]_cons = 0
```

LPRICE	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
LPRICE						
LAREA	.6624072	.055697	11.89	0.000	.5532431	.7715712
LFLOOR	.0375215	.0077354	4.85	0.000	.0223604	.0526827
GARV	.0381015	.0142266	2.68	0.007	.0102178	.0659852
INDV	-.07346	.0282395	-2.60	0.009	-.1288083	-.0181116
SEAVF	.0504307	.0223033	2.26	0.024	.006717	.0941443
SEAVS	-.0448535	.025468	-1.76	0.078	-.0947699	.0050628
SEAVM	.0264926	.0189863	1.40	0.163	-.01072	.0637051
MONV	-.0146642	.0193756	-0.76	0.449	-.0526396	.0233112
STRI	.0224552	.1480547	0.15	0.879	-.2677267	.3126372
STRN	.0311808	.1227643	0.25	0.800	-.2094327	.2717943
UNLUCK	-.0193196	.0210785	-0.92	0.359	-.0606327	.0219936
_cons	3.725121	.3871267	9.62	0.000	2.966367	4.483875

11. We have redefined the variable label of UNLUCK because this variable does not depend on whether an apartment belongs to block 4 as indicated in Magnus et al. (2011). The variable BLOCK, which was ignored in this study, is provided separately.

```
lnsigma2
        STRN   -4.319087   .0758102   -56.97   0.000    -4.467672   -4.170501
       _cons          0  (omitted)
```

```
Wald test of lnsigma2=0: chi2(1) = 3245.85                Prob > chi2 = 0.0000
```

```
. hetwals `y´ (`X1´) `X2´, het(`V´) hconstraints(1) `hetwals_ops´
```

```
Heteroskedastic WALS estimates                    Number of obs =     560
Prior : laplace                                   Residual df   =     548
Wald test for heteroskedasticity                  k1            =       6
chi2(1) = 3245.85                                 k2            =       6
Prob > chi2 = 0.0000                              MC reps       =    1000
```

LPRICE	Coef.	DS Bias	DS Std.Err.	DS RMSE	MC-DS [95% Conf.	Int.]
focus						
LAREA	.650849	-.007273	.040142	.040796	.569395	.742907
LFLOOR	.038796	.000909	.00606	.006128	.026212	.050199
GARV	.041172	.002206	.011065	.011282	.016199	.061089
INDV	-.074521	-.000775	.02221	.022223	-.116263	-.030022
SEAVF	.049547	.000022	.016271	.016271	.015098	.085101
auxiliary						
SEAVS	-.034444	.008913	.017139	.019318	-.080899	-.003749
SEAVM	.018752	-.005539	.012042	.013254	-.006478	.055558
MONV	-.008963	.003004	.010657	.011072	-.042239	.013537
STRI	.014383	-.00449	.078191	.07832	-.187171	.219183
STRN	.017993	-.006188	.065084	.065378	-.144251	.199955
UNLUCK	-.012216	.003958	.01168	.012332	-.047487	.014789
focus						
_cons	3.80858	.050043	.273685	.278222	3.17784	4.36246

The ML results (point estimates and SEs) coincide with those reported in Magnus et al. (2011, Table 2). Except for a small difference in the constant term, the WALS point estimates are also the same but the plug-in estimates of the SEs are substantially smaller than those reported in this previous study. The discrepancies in the SE reflect an important theoretical issue on the frequentist properties of the WALS estimator. In addition to ignoring the bias of the posterior mean in the normal location model, earlier WALS studies estimated the sampling variance of the posterior mean by the posterior variance. However, as shown more recently by De Luca et al. (2022), this is not correct, because the posterior variance represents a first-order approximation of the sampling *standard deviation* (not the sampling variance) of the posterior mean. The new plug-in estimators of the sampling moments address these issues and yield refinements that are more accurate than first-order approximations. In this example, shrinkage of the six auxiliary parameters in the mean function introduces a small estimation bias that is more than offset by a smaller variance. This bias-precision trade-off represents the essence of the improved MSE performance of model-averaging estimators. Here, the efficiency gains occur for all parameters and are also visible from the confidence interval lengths. For example, in contrast to the ML results, the 95% WALS confidence interval for the effect of `SEAVS` does not include zero.

Since the consistency of the GLS estimates relies crucially on the correct specification of the variance function, we now extend the model of Magnus et al. (2011) in three directions. First, we remove the constraint placed on the intercept of the variance function that corresponds to restricting the variance of the outcome to be equal to one for all apartments with `STRN=0`. Second, we consider a larger model that imposes no exclusion restrictions among the regressors of the mean and variance functions. Third, we also introduce in the model a set of binary indicators for the 34 blocks of the South Horizon estate. The ML estimates of our larger model are:

```
. hetregress `y´ `X´ i.BLOCK, het(`X´ i.BLOCK) `hetregr_ops´
Heteroskedastic linear regression               Number of obs     =        560
ML estimation
                                                Wald chi2(44)     =     887.63
Log likelihood =   535.266                      Prob > chi2       =     0.0000
```

LPRICE	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
LPRICE						
LAREA	.5485777	.0554074	9.90	0.000	.4399811	.6571742
LFLOOR	.0303352	.0058966	5.14	0.000	.018778	.0418924
GARV	.0515203	.0121301	4.25	0.000	.0277458	.0752948
INDV	-.0817373	.0332082	-2.46	0.014	-.1468242	-.0166505
SEAVF	.0388607	.0202203	1.92	0.055	-.0007703	.0784917
SEAVS	-.0252818	.0140623	-1.80	0.072	-.0528433	.0022798
SEAVM	.0151497	.0180198	0.84	0.401	-.0201685	.0504679
MONV	.0051382	.0131557	0.39	0.696	-.0206464	.0309228
STRI	.0243226	.0226031	1.08	0.282	-.0199786	.0686239
STRN	.0559876	.0218259	2.57	0.010	.0132097	.0987656
UNLUCK	-.027237	.0193075	-1.41	0.158	-.0650791	.0106051
BLOCK						
2	-.023447	.0623375	-0.38	0.707	-.1456262	.0987323
(output omitted)						
34	-.0197855	.0585189	-0.34	0.735	-.1344804	.0949093
_cons	4.51045	.3741602	12.05	0.000	3.77711	5.243791
lnsigma2						
LAREA	2.473161	.8449543	2.93	0.003	.8170814	4.129241
LFLOOR	-.0373737	.0918521	-0.41	0.684	-.2174005	.142653
GARV	.0973304	.2335433	0.42	0.677	-.3604061	.5550669
INDV	.2645547	.3994216	0.66	0.508	-.5182971	1.047407
SEAVF	-.1192117	.3193888	-0.37	0.709	-.7452022	.5067787
SEAVS	.0569094	.2627662	0.22	0.829	-.4581029	.5719217
SEAVM	.0975122	.3344496	0.29	0.771	-.5579971	.7530215
MONV	.3061975	.22957	1.33	0.182	-.1437515	.7561464
STRI	1.082571	.4010004	2.70	0.007	.2966249	1.868517
STRN	.6364127	.3972803	1.60	0.109	-.1422424	1.415068
UNLUCK	.6440932	.2651985	2.43	0.015	.1243136	1.163873
BLOCK						
2	-.7177862	.5329064	-1.35	0.178	-1.762263	.3266912
(output omitted)						
34	-3.065102	.8349262	-3.67	0.000	-4.701528	-1.428677
_cons	-20.19457	5.560692	-3.63	0.000	-31.09333	-9.295814

```
Wald test of lnsigma2=0: chi2(44) = 95.94                 Prob > chi2 = 0.0000
. predict ml_rmse, stdp
. forvalue b=2(1)34 {
  2.         local X_B `X_B´ `b´.BLOCK
  3. }
. test   [LPRICE]: `X_B´
 ( 1)  [LPRICE]2.BLOCK = 0
  (output omitted)
 (33)  [LPRICE]34.BLOCK = 0
           chi2( 33) =   91.09
         Prob > chi2 =    0.0000
```

These results reveal that the model of Magnus et al. (2011) neglects relevant sources of heterogeneity in the mean and variance functions. In particular, the intercept of the variance function is strongly significant (also in their original model), as are most of the coefficients associated with the indicators for the blocks and those associated with the regressors `LAREA`, `STRI`, and `UNLUCK`. Moreover, although none of the individual indicators for the blocks has a significant effect on the mean of `LPRICE` at the 5% level, a Wald test strongly rejects the hypothesis that their coefficients are jointly insignificant. Our larger model yields considerably different ML estimates of the mean function. For example, the elasticity of the apartment sales prices per square foot with respect to its size in square feet reduces from 0.662 with a 95% confidence interval of $[0.553, 0.772]$ to 0.549 with a 95% confidence interval of $[0.440, 0.657]$. In addition, `SEAVF` has a relatively smaller effect that is no longer significant at the 5% level, while `STRN` has a relatively larger effect that is now significant at the 5% level.

In WALS, we do not change the original partition of the focus and auxiliary regressors on the basis of the new ML estimates. However, we still need to decide whether to treat the 33 additional indicators for the blocks as focus regressors or as auxiliary regressors. This choice does not affect the variance function, whose estimates are always taken from the unrestricted ML estimates, but it may have implications for the WALS estimates of the mean function. Below, we investigate this issue by comparing the WALS estimates resulting from these two models:

```
. hetwals `y´ (`X1´ i.BLOCK) `X2´ , het(`X´ i.BLOCK) `hetwals_ops´
Heteroskedastic WALS estimates                    Number of obs =     560
Prior : laplace                                   Residual df   =     515
Wald test for heteroskedasticity                  k1            =      39
chi2(44) = 95.94                                  k2            =       6
Prob > chi2 = 0.0000                              MC reps       =    1000
```

LPRICE	Coef.	DS Bias	DS Std.Err.	DS RMSE	MC-DS [95% Conf.	Int.]
focus						
LAREA	.543516	-.005367	.039111	.039477	.467282	.62823
LFLOOR	.03108	.000558	.005072	.005102	.020873	.040661
GARV	.05637	.003958	.009903	.010665	.031042	.0736
INDV	-.081007	.000331	.02511	.025112	-.132392	-.02939
SEAVF	.042959	.004371	.017336	.017878	.003504	.075647

```
   2.BLOCK |  -.027402   -.002674    .060311    .06037   -.143528    .091313
  (output omitted)
  34.BLOCK |  -.033914   -.011401    .056724    .057859   -.13392    .084611
-----------+------------------------------------------------------------------
auxiliary  |
     SEAVS |    -.0191    .005153    .010348    .01156   -.046896   -.000217
     SEAVM |   .009264   -.003321    .011156    .01164   -.015834    .042727
      MONV |   .004016   -.000978    .007744    .007806   -.017514     .02396
      STRI |   .020284    -.00419     .01232    .013013   -.007906     .05518
      STRN |    .04632   -.009522    .015489    .018182    .020827    .089494
    UNLUCK |  -.017916      .0056    .012714    .013893   -.057179     .01003
-----------+------------------------------------------------------------------
focus      |
     _cons |   4.55213    .042742     .26265    .266106    3.97097    5.06835
------------------------------------------------------------------------------

. predict wals1_rmse, rmsep

. hetwals `y´ (`X1´) `X2´ i.BLOCK , het(`X´ i.BLOCK) `hetwals_ops´

Heteroskedastic WALS estimates                    Number of obs =       560
Prior : laplace                                   Residual df   =       515
Wald test for heteroskedasticity                  k1            =         6
chi2(44) = 95.94                                  k2            =        39
Prob > chi2 = 0.0000                              MC reps       =      1000
------------------------------------------------------------------------------
             |                 DS         DS         DS            MC-DS
      LPRICE |      Coef.      Bias   Std.Err.       RMSE      [95% Conf. Int.]
-------------+----------------------------------------------------------------
focus        |
       LAREA |    .538568  -.006244    .037923    .038434     .46667    .621147
      LFLOOR |    .030432   .000397    .004981    .004997    .020052    .040109
        GARV |     .05428   .002149    .009745    .009979     .03296    .071978
        INDV |   -.079495  -.001018    .021701    .021725   -.123651   -.023189
       SEAVF |    .053023   .011484    .016795    .020346    .007522    .081615
-------------+----------------------------------------------------------------
auxiliary    |
       SEAVS |   -.020189   .004849    .010978    .012001   -.048679    .000623
       SEAVM |    .012197  -.003217     .01262    .013023   -.015019    .044965
        MONV |    .006244  -.000654    .008119    .008145   -.016149    .026431
        STRI |     .02291  -.003344    .012834    .013262   -.006561     .05769
        STRN |    .048574  -.007958    .015746    .017643    .022547    .089749
      UNLUCK |   -.018019   .005576    .012767    .013932   -.058091    .010654
     2.BLOCK |   -.010039   .005602    .044232    .044585   -.130818    .106857
  (output omitted)
    34.BLOCK |   -.00363    .005864    .042382    .042786   -.112824    .102277
-------------+----------------------------------------------------------------
focus        |
       _cons |   4.56067     .03585    .254329    .256843    3.99435    5.06153
------------------------------------------------------------------------------

. predict wals2_rmse, rmsep

. sum ml* wals*

    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
     ml_rmse |        560    .0276952    .0078066   .0170713   .0578341
  wals1_rmse |        560    .0254138    .0070759    .016333   .0541742
  wals2_rmse |        560    .0217798    .0053496   .0124492   .0441208
```

Both models yield point estimates of the focus and auxiliary parameters in the mean function that have the same signs as and are close to the ML estimates. In the first

model, where the dummy variables in `i.BLOCK` are treated as focus regressors, all WALS estimates show better RMSE performance than the ML estimates. In the second model, where the dummy variables in `i.BLOCK` are treated as auxiliary regressors, we find additional RMSE gains in the estimated coefficients of `INDV` and `i.BLOCK` and substantially larger bias and RMSE in the estimated coefficient of `SEAVF`. The average RMSE of the predicted values is equal to 0.028 for the ML estimates, 0.025 for the WALS estimates of the first model, and 0.022 for the WALS estimates of the second model. Using the `plugin(ml)` option of `hetwals` (i.e., the plug-in ML estimators of the sampling moments instead of the default plug-in double-shrinkage estimators), we see that the average RMSE of the WALS predictions becomes 0.026 for the first model and 0.023 for the second model. Similarly, using the `prior(`*priorname*`)` option, we see that the WALS estimates are also robust with respect to the choice of the prior distribution. Even though the Laplace prior is not robust, this result is not surprising because most of the t-ratios stored in `e(t_ratios)` (i.e., the $k_2 \times 1$ vector of observed t-ratios on the auxiliary coefficients of the transformed model) are not particularly large (the maximum t-ratio is 4.530). Priors with a high degree of robustness such as `pareto`, `horseshoe`, `log` and `cauchy` become particularly effective in cases when these t-ratios are large.

8 Attrition between the first two waves of SHARE

Next we use the `glmwals` command to study attrition in the first two waves of the Survey of Health, Aging and Retirement in Europe (SHARE). For comparability with the earlier study of De Luca et al. (2018), the data are extracted from Release 5.0 of the SHARE archive.[12] The sample consists of 18,092 household respondents from 11 European countries who have already participated in the baseline wave of SHARE. The dependent variable is the binary indicator `Part`, which takes the value 1 if the respondent also agrees to participate in the second wave, and 0 otherwise. Our goal is to study the determinants of the participation probability in the second wave, conditional on participation in the first wave. De Luca et al. (2018) focused on the following predictors:

```
. use "DMP_2018", clear
. describe
Contains data from DMP_2018.dta
 Observations:        18,092
    Variables:            18                  26 Jun 2024 17:33
                                              (_dta has notes)
```

Variable name	Storage type	Display format	Value label	Variable label
mergeid	str12	%12s		Person identifier
country	byte	%14.0g	country	Country identifier
Part	byte	%28.0g	noyes	1 if participates in wave 2; 0 otherwise
Age	byte	%9.0g		Age of household respondent in

12. The latest public release of SHARE data is Release 9.0. Here, we use Release 5.0 only to replicate the WALS estimates presented in De Luca et al. (2018, Table 2). For details on the sampling design and fieldwork procedures of SHARE, see Malter and Börsch-Supan (2015).

```
                                                  2004
Female          byte    %9.0g       sex           1 if female; 0 otherwise
Couple          byte    %14.0g      noyes         1 if lives with a partner; 0
                                                  otherwise
BigCity         byte    %9.0g                     1 if lives in a big city; 0
                                                  otherwise
HEduc           byte    %9.0g                     1 if high education; 0 otherwise
Employed        byte    %9.0g                     1 if employed; 0 otherwise
GoodSRH         byte    %9.0g                     1 if good health; 0 otherwise
Doctor          byte    %9.0g                     Number of visits to medical doctor
EuroD           byte    %9.0g                     Euro-D depression index
Recall          byte    %9.0g                     Score of recall tests
Fluency         byte    %9.0g                     Score of fluency tests
AgeP            byte    %9.0g                     Age of partner in 2004 (0 if
                                                  couple=0)
SocAct          byte    %9.0g                     Number of social activities
FemaleIV        byte    %9.0g       sex           1 if female interviewer; 0
                                                  otherwise
AgeIV           byte    %9.0g                     Age of interviewer in 2004
```

Below, we transform and partition the regressors as in De Luca et al. (2018) to reproduce their iterative WALS estimates of a logit model for the participation probability of French households:

```
. qui gen CAge=Age-50
. qui gen CAge2=CAge^2/10
. qui gen FemCAge=Female * CAge
. qui gen FemCAge2=Female * CAge^2/10
. qui gen CAgeP=AgeP-50   if Couple!=0
. qui replace CAgeP=0     if Couple==0
. qui gen CAgeIV=AgeIV-50
. qui gen CDoctor=Doctor-5
. qui gen CEuroD=EuroD-3
. qui gen CRecall=Recall-9
. qui gen CSocAct=SocAct-1
. local y  "Part"
. local X1 "CAge CAge2 Female FemCAge FemCAge2 Couple HEduc BigCity Employed"
. local X2 "GoodSRH CDoctor CEuroD CRecall CSocAct CAgeP FemaleIV CAgeIV"
. local opts "family(binomial) prior(Weibull) rseed(123)"
. glmwals `y´ (`X1´) `X2´ if country==17, `opts´

Iteration 1:  Rel. Diff. e(b) initialized
Iteration 2:  Rel. Diff. e(b) =   .000795
Iteration 3:  Rel. Diff. e(b) =  .0000158
Iteration 4:  Rel. Diff. e(b) =  2.40e-07

Iterative WALS estimates of GLM                    Number of obs =    1822
Prior             : weibull                        Residual df   =    1804
Family            : Bernoulli                      k1            =      10
Link              : Logit                          k2            =       8
Variance function : V(u) = u*(1-u)                 sigma         =       1
Link function     : g(u) = ln(u/(1-u))             MC reps       =    1000

------------------------------------------------------------------------------
             |                 DS        DS        DS            MC-DS
```

Part	Coef.	Bias	Std.Err.	RMSE	[95% Conf.	Int.]
focus						
CAge	.032031	-.004272	.03105	.031342	-.026808	.0969
CAge2	-.004742	-.000195	.0091	.009103	-.022228	.013001
Female	-.083268	-.013162	.250394	.25074	-.568622	.414939
FemCAge	.042082	.00036	.038177	.038179	-.032919	.113418
FemCAge2	-.015696	.000414	.011688	.011695	-.038499	.006583
Couple	-.00335	-.059703	.158945	.169788	-.299029	.381162
HEduc	.300702	.02568	.114502	.117346	.048613	.516371
BigCity	-.210946	-.003056	.105846	.10589	-.422837	-.013756
Employed	-.115539	.009713	.158348	.158646	-.439878	.215621
auxiliary						
GoodSRH	.158367	-.049292	.098206	.109883	-.019039	.443103
CDoctor	-.007452	.002443	.005523	.006039	-.024541	.004221
CEuroD	.039184	-.01214	.018157	.021842	.002441	.096762
CRecall	.037098	-.009416	.016069	.018624	.007776	.083471
CSocAct	.116514	-.028199	.051902	.059068	.028028	.257116
CAgeP	-.016953	.004738	.00757	.00893	-.039224	-.002857
FemaleIV	.032754	-.011469	.079944	.080762	-.184249	.253926
CAgeIV	-.01596	.003945	.006349	.007475	-.032998	-.004785
focus						
_cons	.513604	.055969	.294168	.299445	-.140334	1.07283

Based on the same estimation options, including the default logit link of the binomial family, the starting values, and the Weibull prior, our iterative WALS estimates converge in 4 iterations as indicated in De Luca et al. (2018, p. 9). The point estimates of the focus parameters also coincide with those reported in the last column of their Table 2, but not the SEs, because `glmwals` calculates the plug-in estimates of the sampling moments (see the discussion in Section 7).[13] Notice that, although the data contain 1,932 French households, the estimation sample was restricted to 1,822 observations because of missing values in the regressors. The confidence intervals reveal that only 2 focus regressors (`HEduc` and `BigCity`) are significant at the 5% level. On the other hand, there are 5 out of 8 auxiliary regressors (`CEuroD`, `CRecall`, `CSocAct`, `CAgeP` and `CAgeIV`) that have a significant effect on the participation probability. As usual, the interpretation of the estimated coefficients is complicated by the nonlinear nature of the model and the presence of polynomial and interaction terms. However, if we specify the regressors through the factor notation, then we can exploit the `margwals` command, which will make the interpretation of our estimation results easier and more transparent. This is what we do below.

De Luca et al. (2018) used this model to compare the WALS estimates of gender-specific age profiles of the participation probability with those obtained from classical and penalized ML methods, and other strictly frequentist and strictly Bayesian model-averaging procedures. In what follows, we compute the iterative WALS estimates of a much larger model to highlight the computational performance of this model-averaging

13. Similarly, one can reproduce the one-step WALS estimates in the second to last two columns of De Luca et al. (2018, Table 2) by specifying, respectively, the options `onestep` and `init(irls_r)` and the options `onestep` and `init(irls_u)`.

method. Suppose we want to investigate the effects of individual cognitive abilities, as measured by the score on the recall test performed in the first wave, on the probability of participating in the second wave. Instead of restricting our attention to France, we pool data from 11 countries and include a set of country dummies and their interactions with the key variables of interest to account for cross-country heterogeneity. Specifically, we redefine the focus and auxiliary regressors as follows:

```
. #delimit;
. local X1 "c.Recall i.country c.Age i.Female i.Couple i.HEduc i.BigC i.Empl";
. local X2 "i.GoodSRH c.Doctor c.EuroD c.SocAct c.AgeP i.FemaleIV c.AgeIV
>     c.Recall#c.Recall i.country#c.Recall i.country#c.Recall#c.Recall
>     c.Age#c.Age i.Female#c.Age i.Female#c.Age#c.Age
>     i.country#c.Age i.country#i.Female i.country#c.Age#i.Female
>     i.country#c.Age#c.Age i.country#i.Female#c.Age#c.Age";
. #delimit cr
```

In total, this model contains 18 focus regressors and 81 auxiliary regressors. The focus regressors include the score on the recall test, basic socio-demographic characteristics of the respondent, the country dummies, and a constant term; while the auxiliary regressors include other controls for physical and mental health and social activities of the respondent, demographic characteristics of the partner and the interviewer, plus quadratic terms and interactions of `Recall` with `country` and `Age` with `Female` and `country`. As an example of a link function which is not canonical, we also consider a probit model. Furthermore, given the large sample size, we now use the `log` prior to ensure that the WALS estimation bias becomes negligible for large values of the population t-ratios in the transformed model. The iterative WALS estimates of our probit model are:

```
. local opts "fam(b) link(p) prior(log) rseed(123) sav(WREPS,replace) notab"
. glmwals `y´ (`X1´) `X2´, `opts´

Iteration 1:  Rel. Diff. e(b) initialized
Iteration 2:  Rel. Diff. e(b) =  .0495148
Iteration 3:  Rel. Diff. e(b) =  .0005879
Iteration 4:  Rel. Diff. e(b) =  .0000329
Iteration 5:  Rel. Diff. e(b) =  7.43e-07

Iterative WALS estimates of GLM                     Number of obs =   17051
Prior             : log                             Residual df   =   16952
Family            : Bernoulli                       k1            =      18
Link              : Probit                          k2            =      81
Variance function : V(u) = u*(1-u)                  sigma         =       1
Link function     : g(u) = invnormal(u)             MC reps       =    1000
```

As before, the estimation procedure converges quickly in 5 iterations. Note that we used the `notable` option to suppress the display of the long coefficient table. Since regressors have been specified in factor notation, we now apply the `margwals` command to compute the country-specific WALS estimates of the average marginal effects of `Recall` on the participation probability:

```
. margwals, dydx(Recall) over(country)
Average marginal effects                Number of obs = 17051
                                        MC reps       = 1000
```

	Coef.	MC-DS [95% Conf.	Interval]
Recall			
Recall:11bn.country	.0129679	.0070382	.0210416
Recall:12.country	.0022684	-.0066919	.0075696
Recall:13.country	.0112511	.0056125	.0185794
Recall:14.country	.0091852	.0025589	.0154486
Recall:15.country	.0090726	.0009196	.0162689
Recall:16.country	.0081618	.0006812	.0147063
Recall:17.country	.0106605	.004993	.0168367
Recall:18.country	.0144888	.0077992	.0239067
Recall:19.country	-.0016531	-.0095953	.0032749
Recall:20.country	.0159057	.0082799	.0282735
Recall:23.country	.0097567	.0052589	.0150726

This post-estimation task is more computationally demanding than the previous estimation step as it involves computing the simulation-based confidence intervals using the 1000 replications of the bias-corrected WALS estimator stored in `WREPS.dta`.[14] Notice that the WALS confidence intervals are not symmetric around the point estimates of the average marginal effects because the estimator has a finite-sample bias and its distribution is not necessarily normal. We find that, except for Germany and Greece, the average marginal effects of `Recall` on the participation probability are positive and significant at the 5% level. The average marginal effects resulting from the unrestricted ML estimates are:

```
. qui probit `y´ `X1´ `X2´
. margins, dydx(Recall) over(country)
Average marginal effects                                  Number of obs = 17,051
Model VCE: OIM
Expression: Pr(Part), predict()
dy/dx wrt:  Recall
Over:       country
```

	dy/dx	Delta-method std. err.	z	P>|z|	[95% conf.	interval]
Recall						
country						
Austria	.0136654	.0040228	3.40	0.001	.0057808	.0215499
Germany	-.0006411	.0036962	-0.17	0.862	-.0078855	.0066033
Sweden	.013506	.0037812	3.57	0.000	.006095	.0209169
Netherlands	.0088492	.003714	2.38	0.017	.0015699	.0161285
Spain	.00794	.0046645	1.70	0.089	-.0012022	.0170823
Italy	.0068598	.0041787	1.64	0.101	-.0013303	.0150499
France	.0106645	.0035821	2.98	0.003	.0036438	.0176852
Denmark	.018871	.0043025	4.39	0.000	.0104382	.0273038
Greece	-.0039357	.0030269	-1.30	0.194	-.0098683	.0019969
Switzerland	.0208797	.0053457	3.91	0.000	.0104023	.031357

14. Using a ThinkPad laptop with processor 11th Gen Intel(R) Core(TM) i7-11800H, 32 GB of RAM and Stata/MP8 (version 18.0), these calculations were performed in around 10 minutes. Estimation is instead executed in a few seconds.

Belgium	.0109402	.0029707	3.68	0.000	.0051177	.0167627

In Germany and Greece, the average marginal effects of `Recall` on the participation probability are again insignificant at the conventional levels. In Austria, France and The Netherlands, the relative differences between WALS and unrestricted ML estimates are smaller than 5% (in absolute value). In the other countries, the differences are more sizeable: 11% in Belgium, 14% in Spain, 17% in Sweden, 19% in Italy, 23% in Denmark, and 24% in Switzerland. Excluding Germany and Greece, we also find that ML leads to relatively larger confidence intervals. In particular, the ML estimates of the average marginal effects of `Recall` on the participation probability of Spanish and Italian households are far from being significant at the 5% level.

9 Conclusions

This paper has presented four new Stata commands for WALS estimation of statistical models that go beyond the classical linear regression model. Our aim in this paper has been to enlarge the class of models that can be considered in empirical applications of this model-averaging technique. The current version of the WALS package covers linear models with either i.i.d. errors, multiplicative heteroskedasticity, or stationary AR(1) errors, fixed-effects and random-effects panel-data models with either i.i.d. or AR(1) errors, and univariate GLMs. Of course, the WALS package is a living organism, and we expect further extensions and improvements to be made in the future. For example, FGLS transformations for nonspherical errors could be further extended to cover other relevant scenarios. Similarly, the WALS approach to GLMs could be extended to negative binomial models with an unknown overdispersion parameter as recently shown by Huynh (2024).

10 Acknowledgments

Giuseppe De Luca acknowledges financial support by the European Union - Next Generation EU, in the framework of the GRINS - Growing Resilient, Inclusive and Sustainable Project (GRINS PE00000018 – CUP E63C22002140007). The views and opinions expressed are solely those of the authors and do not necessarily reflect those of the European Union, and the European Union cannot be held responsible for them. We thank Xinyu Zhang for providing us with the dataset used in Section 7. The paper also uses data from SHARE Waves 1 and 2 (DOI: 10.6103/SHARE.w5.500), Release 5.0. The SHARE data collection has been funded by the European Commission, DG RTD through FP5 (QLK6-CT-2001-00360), FP6 (SHARE-I3: RII-CT-2006-062193, COMPARE: CIT5-CT-2005-028857, SHARELIFE: CIT4-CT-2006-028812), FP7 (SHARE-PREP: GA N. 211909, SHARE-LEAP: GA N. 227822, SHARE M4: GA N. 261982, DASISH: GA N. 283646), and Horizon 2020 (SHARE-DEV3: GA N. 676536, SHARE-COHESION: GA N. 870628, SERISS: GA N. 654221, SSHOC: GA N. 823782, SHARE-COVID19: GA N. 101015924); and by DG Employment, Social Affairs & Inclusion

through VS 2015/0195, VS 2016/0135, VS 2018/0285, VS 2019/0332, VS 2020/0313 and SHARE-EUCOV: GA N. 101052589 and EUCOVII: GA N. 101102412. Additional funding from the German Ministry of Education and Research, the Max Planck Society for the Advancement of Science, the U.S. National Institute on Aging (U01_AG09740-13S2, P01_AG005842, P01_AG08291, P30_AG12815, R21_AG025169, Y1-AG-4553-01, IAG_BSR06-11, OGHA_04-064, BSR12-04, R01_AG052527-02, HHSN2712-01300071C, RAG052527A), and from various national funding sources is gratefully acknowledged (see www.share-eric.eu).

11 Programs and supplemental material

To install the software files as they existed at the time of publication of this article, type

```
. net sj XX
. net install stXXX (to install program files, if available)
. net get stXXX (to install ancillary files, if available)
```

12 References

Baltagi, B. H. 2021. *Econometric Analysis of Panel Data*. 6th ed. Switzerland: Springer.

Baltagi, B. H., and Q. Li. 1991. A transformation that will circumvent the problem of autocorrelation in an error-component model. *Journal of Econometrics* 48: 385–393.

Baltagi, B. H., and P. X. Wu. 1999. Unequally spaced panel data regressions with AR(1) disturbances. *Econometric Theory* 15: 814–823.

Bhargava, A., L. Franzini, and W. Narendranathan. 1982. Serial correlation and the fixed effects model. *Review of Economic Studies* 49: 533–549.

Davidson, R., and J. G. MacKinnon. 1993. *Estimation and Inference in Econometrics*. New York: Oxford University Press.

De Luca, G., and J. R. Magnus. 2024. Weighted-average least squares estimation: Improvements and extensions. The Stata Journal (under review).

———. 2025. Weighted-average least squares estimation of panel-data models. In *Advances in Shrinkage and Penalized Estimation Strategies: Honoring the Contributions of A. K. Md. Ehsanes Saleh*, ed. M. Arashi and M. Norouzirad, 00–00. New York: Springer Series "Emerging Topics in Statistics and Biostatistics.".

De Luca, G., J. R. Magnus, and F. Peracchi. 2018. Weighted-average least squares estimation of generalized linear models. *Journal of Econometrics* 204: 1–17.

———. 2022. Sampling properties of the Bayesian posterior mean with an application to WALS estimation. *Journal of Econometrics* 230: 299–317.

———. 2023. Weighted-average least squares (WALS): Confidence and prediction intervals. *Computational Economics* 61: 1637–1664.

———. 2025. Bayesian estimation of the normal location model: A non-standard approach. *Oxford Bulletin of Economics and Statistics* Forthcoming.

Goldberger, A. S. 1962. Best linear unbiased prediction in the generalized linear regression model. *Journal of the American Statistical Association* 57: 369–375.

Harvey, A. C. 1976. Estimating regression models with multiplicative heteroscedasticity. *Econometrica* 44: 461–465.

Hildreth, C., and J. Y. Lu. 1960. Demand relations with autocorrelated disturbances Reprinted in Agricultural Experiment Station Technical Bulletin, No. 276. East Lansing, MI: Michigan State University Press.

Huynh, K. 2024. Weighted-average least squares for negative binomial regression. ArXiv preprint arXiv:2404.11324.

Judge, G. G., W. E. Griffiths, R. C. Hill, H. Lütkepohl, and T.-C. Lee. 1985. *The Theory and Practice of Econometrics*. New York: Wiley.

Magnus, J. R., and G. De Luca. 2016. Weighted-average least squares: A review. *Journal of Economic Surveys* 30: 117–148.

Magnus, J. R., O. Powell, and P. Prüfer. 2010. A comparison of two model averaging techniques with an application to growth empirics. *Journal of Econometrics* 154: 139–153.

Magnus, J. R., A. T. K. Wan, and X. Zhang. 2011. Weighted average least squares estimation with nonspherical disturbances and an application to the Hong Kong housing market. *Computational Statistics & Data Analysis* 55: 1331–1341.

Magnus, J. R., W. Wang, and X. Zhang. 2016. Weighted-average least squares prediction. *Econometric Reviews* 35: 1040–1074.

Malter, F., and A. Börsch-Supan. 2015. *SHARE Wave 5: Innovations & Methodology*. Munich: MEA, Max Planck Institute for Social Law and Social Policy.

McCullagh, P., and J. A. Nelder. 1989. *Generalized Linear Models*. 2nd ed. London: Chapman and Hall.

Mundlak, Y. 1978. On the pooling of time series and cross section data. *Econometrica* 46: 69–85.

Nelder, J. A., and R. W. M. Wedderburn. 1972. Generalized linear models. *Journal of the Royal Statistical Society (Series A)* 135: 370–384.

About the authors

Giuseppe De Luca is full professor of econometrics at the Department of Economics, Business and Statistics (SEAS) of the University of Palermo.

Jan R. Magnus is emeritus professor of econometrics at Tilburg University, and extraordinary professor at the Vrije Universiteit Amsterdam, both in The Netherlands.

Appendix A: Stored results

This appendix lists all results stored by the new WALS commands: `hetwals` in Appendix A.1, `ar1wals` in Appendix A.2, `xtwals` in Appendix A.3, and `glmwals` in Appendix A.4. In each case, results on the sampling moments and the confidence intervals are not available if one specifies the `fast` option.

Appendix A.1: Results stored by `hetwals`

`hetwals` stores the following in `e()`:

Scalars

`e(N)`	number of observations
`e(k0)`	number of columns of `e(b)` (including base levels of factor variables)
`e(k1)`	number of (not omitted) focus regressors (mean function)
`e(k2)`	number of (not omitted) auxiliary regressors (mean function)
`e(rank)`	rank of `e(V)` (also `e(k1)+e(k2)`)
`e(df_r)`	residual degrees of freedom (mean function)
`e(quadnpts)`	number of quadrature points for `gauss` method
`e(quadatol)`	absolute tolerance for `adaptive` quadrature method
`e(quadrtol)`	relative tolerance for `adaptive` quadrature method
`e(reps)`	number of MC replications for confidence intervals
`e(level)`	confidence level
`e(hchi2)`	chi-squared test for homoskedasticity
`e(hchi2_df)`	degrees of freedom of `e(hchi2)`
`e(hchi2_p)`	p-value of `e(hchi2)`

Macros

`e(cmdline)`	command as typed
`e(cmd)`	`hetwals`
`e(title)`	title appearing in header
`e(depvar)`	name of dependent variable
`e(allvars)`	names of all regressors (mean function)
`e(omitvars)`	names of omitted regressors (mean function)
`e(focvars)`	names of (not omitted) focus regressors (mean function)
`e(auxvars)`	names of (not omitted) auxiliary regressors (mean function)
`e(constype)`	type of constant term in the mean function: `noconstant`, `focus`, `auxiliary`
`e(hvars)`	names of regressors in the variance function
`e(hmethod)`	estimation method for the variance function: `ml`, `twostep`
`e(htest)`	type of test for `e(hchi2)`: LR, Wald
`e(wtype)`	weight type (if specified)
`e(wexp)`	weight expression (if specified)
`e(prior)`	prior distribution for the population t-ratios
`e(quadmethod)`	quadrature method: `gauss`, `adaptive`, `analytic`
`e(plugin)`	estimate of sampling moments: `ds`, `ml`
`e(bcsimdata)`	file of MC replications of the bias-corrected WALS estimator
`e(properties)`	`b V`
`e(predict)`	program to implement `predict`
`e(marginsok)`	predictions allowed by `margins` and `margwals`

Matrices

`e(b)`	estimated parameter vector of the mean function
`e(V)`	estimated variance matrix of `e(b)`
`e(bias)`	estimated bias vector of `e(b)`
`e(rmse)`	estimated RMSE vector of `e(b)`
`e(MSE)`	estimated MSE matrix of `e(b)`
`e(ci)`	matrix of confidence intervals (mean function)
`e(hb)`	estimated parameter vector of the variance function
`e(hV)`	estimated variance matrix of `e(hb)`
`e(t_ratios)`	$k_2 \times 1$ vector of observed t-ratios in the normal location model
`e(wals_pm)`	$k_2 \times 1$ vector of posterior means in the normal location model
`e(wals_wgt)`	$k_2 \times 1$ vector of WALS weights (diagonal elements of matrix Λ)
`e(priorpar)`	vector of prior parameters

Functions

`e(sample)`	marks estimation sample

Appendix A.2: Results stored by `ar1wals`

`ar1wals` stores the following in `e()`:

Scalars

`e(N)`	number of observations
`e(k0)`	number of columns of `e(b)` (including base levels of factor variables)
`e(k1)`	number of (not omitted) focus regressors
`e(k2)`	number of (not omitted) auxiliary regressors
`e(rank)`	rank of `e(V)` (also `e(k1)+e(k2)`)
`e(df_r)`	residual degrees of freedom
`e(sigma)`	estimate of σ_ϵ in the WALS regression on the transformed data
`e(rho)`	first-step estimate of ρ
`e(dw_0)`	Durbin-Watson statistic in the (original) unrestricted model
`e(dw)`	Durbin-Watson statistic in the (transformed) unrestricted model
`e(quadnpts)`	number of quadrature points for `gauss` method
`e(quadatol)`	absolute tolerance for `adaptive` quadrature method
`e(quadrtol)`	relative tolerance for `adaptive` quadrature method
`e(reps)`	number of MC replications for confidence intervals
`e(level)`	confidence level

Macros

`e(cmdline)`	command as typed
`e(cmd)`	`ar1wals`
`e(title)`	title appearing in header
`e(depvar)`	name of dependent variable
`e(allvars)`	names of all regressors
`e(omitvars)`	names of omitted regressors
`e(focvars)`	names of (not omitted) focus regressors
`e(auxvars)`	names of (not omitted) auxiliary regressors
`e(constype)`	type of constant term: `noconstant`, `focus`, `auxiliary`
`e(rhotype)`	method specified in the `rhotype` option
`e(rhomethod)`	estimation method for ρ: `twostep`, `iterated`, `SSE search`
`e(tranmeth)`	transformation method: `corc`, `prais`
`e(prior)`	prior distribution for the population t-ratios
`e(quadmethod)`	quadrature method: `gauss`, `adaptive`, `analytic`
`e(plugin)`	estimate of sampling moments: `ds`, `ml`
`e(bcsimdata)`	file of MC replications of the bias-corrected WALS estimator
`e(properties)`	`b V`
`e(predict)`	program to implement `predict`
`e(marginsok)`	predictions allowed by `margins` and `margwals`

Matrices

`e(b)`	estimated parameter vector
`e(V)`	estimated variance matrix of `e(b)`
`e(bias)`	estimated bias vector of `e(b)`
`e(rmse)`	estimated RMSE vector of `e(b)`
`e(MSE)`	estimated MSE matrix of `e(b)`
`e(ci)`	matrix of confidence intervals
`e(t_ratios)`	$k_2 \times 1$ vector of observed t-ratios in the normal location model
`e(wals_pm)`	$k_2 \times 1$ vector of posterior means in the normal location model
`e(wals_wgt)`	$k_2 \times 1$ vector of WALS weights (diagonal elements of matrix Λ)
`e(priorpar)`	vector of prior parameters

Functions

`e(sample)`	marks estimation sample

Appendix A.3: Results stored by `xtwals`

`xtwals` stores the following in `e()`:

Scalars

`e(N)`	number of observations
`e(n)`	number of units
`e(Tmin)`	minimum number of observations across units
`e(Tavg)`	average number of observations across units
`e(Tmax)`	maximum number of observations across units
`e(Tcon)`	1 if observations are constant across units
`e(k0)`	number of columns of `e(b)` (including base levels of factor variables)
`e(k1)`	number of (not omitted) focus regressors
`e(k2)`	number of (not omitted) auxiliary regressors
`e(rank)`	rank of `e(V)` (also `e(k1)+e(k2)`)
`e(df_r)`	residual degrees of freedom (only if `e(approach)="fe"`)
`e(sigma)`	estimate of σ_ϵ in the WALS regression on the transformed data
`e(sig_nu)`	first-step estimate of σ_ν (only if `e(approach)="re"`)
`e(sig_e)`	first-step estimate of σ_ϵ (only if `e(approach)="re"`)
`e(alpha)`	first-step estimate of α (only if `e(approach)="re"` and `e(Tcon)=1`)
`e(rho)`	first-step estimate of ρ (only if `e(errors)="ar1"`)
`e(quadnpts)`	number of quadrature points for `gauss` method
`e(quadatol)`	absolute tolerance for `adaptive` quadrature method
`e(quadrtol)`	relative tolerance for `adaptive` quadrature method
`e(reps)`	number of MC replications for confidence intervals
`e(level)`	confidence level

Macros

`e(cmdline)`	command as typed
`e(cmd)`	`xtwals`
`e(title)`	title appearing in header
`e(model)`	model setup: `fe-iid`, `re-iid`, `fe-ar1`, `re-ar1`
`e(approach)`	approach: `fe`, `re`
`e(errors)`	regression errors: `iid`, `ar1`
`e(ivar)`	variable denoting units
`e(tvar)`	variable denoting time within units (only if `e(errors)="ar1"`)
`e(depvar)`	name of dependent variable
`e(allvars)`	names of all regressors
`e(omitvars)`	names of omitted regressors
`e(focvars)`	names of (not omitted) focus regressors
`e(auxvars)`	names of (not omitted) auxiliary regressors
`e(constype)`	type of constant term: `focus`, `auxiliary`
`e(wtype)`	weight type (if specified)
`e(wexp)`	weight expression (if specified)
`e(prior)`	prior distribution for the population t-ratios
`e(quadmethod)`	quadrature method: `gauss`, `adaptive`, `analytic`
`e(plugin)`	estimate of sampling moments: `ds`, `ml`
`e(bcsimdata)`	file of MC replications of the bias-corrected WALS estimator
`e(properties)`	`b V`
`e(predict)`	program to implement `predict`
`e(marginsok)`	predictions allowed by `margins` and `margwals`

Matrices

`e(b)`	estimated parameter vector
`e(V)`	estimated variance matrix of `e(b)`
`e(bias)`	estimated bias vector of `e(b)`
`e(rmse)`	estimated RMSE vector of `e(b)`
`e(MSE)`	estimated MSE matrix of `e(b)`
`e(ci)`	matrix of confidence intervals
`e(alpha_stat)`	summary statistics on α_i (only if `e(approach)="re"` and `e(Tcon)=0`)
`e(t_ratios)`	$k_2 \times 1$ vector of observed t-ratios in the normal location model
`e(wals_pm)`	$k_2 \times 1$ vector of posterior means in the normal location model
`e(wals_wgt)`	$k_2 \times 1$ vector of WALS weights (diagonal elements of the matrix Λ)
`e(priorpar)`	vector of prior parameters

Functions

`e(sample)`	marks estimation sample

Appendix A.4: Results stored by `glmwals`

`glmwals` stores the following in `e()`:

Scalars

`e(N)`	number of observations
`e(k0)`	number of columns of `e(b)` (including base levels of factor variables)
`e(k1)`	number of (non omitted) focus regressors
`e(k2)`	number of (non omitted) auxiliary regressors
`e(rank)`	rank of `e(V)` (also `e(k1)+e(k2)`)
`e(df_r)`	residual degrees of freedom
`e(sigma)`	estimated scale parameter
`e(quadnpts)`	number of quadrature points for `gauss` method
`e(quadatol)`	absolute tolerance for `adaptive` method
`e(quadrtol)`	relative tolerance for `adaptive` method
`e(reps)`	number of MC replications for confidence intervals
`e(level)`	confidence level
`e(ic)`	number of iterations
`e(converged)`	1 if converged, 0 otherwise

Macros

`e(cmdline)`	command as typed
`e(cmd)`	`glmwals`
`e(title)`	title appearing in header
`e(depvar)`	name of dependent variable
`e(varfunc)`	program used to calculate the variance function
`e(varfunct)`	variance title
`e(varfuncf)`	variance function
`e(link)`	program used to calculate the link function
`e(linkt)`	link title
`e(linkf)`	link function
`e(btrials)`	number of binomial trials
`e(esttype)`	type of estimates: `one-step`, `iterative`
`e(allvars)`	names of all regressors
`e(omitvars)`	names of omitted regressors
`e(focvars)`	names of (non omitted) focus regressors
`e(auxvars)`	names of (non omitted) auxiliary regressors
`e(constype)`	type of constant term: `noconstant`, `focus`, `auxiliary`
`e(offset)`	linear offset variable
`e(wtype)`	weight type (if specified)
`e(wexp)`	weight expression (if specified)

`e(prior)`	prior distribution for the population t-ratios
`e(quadmethod)`	quadrature method: `gauss`, `adaptive`, `analytic`
`e(plugin)`	estimate of sampling moments: `ds`, `ml`
`e(bcsimdata)`	file of MC replications of the bias-corrected WALS estimator
`e(properties)`	`b V`
`e(predict)`	program used to implement `predict`
`e(marginsok)`	predictions allowed by `margins` and `margwals`

Matrices

`e(b)`	estimated parameter vector
`e(V)`	estimated variance matrix of `e(b)`
`e(bias)`	estimated bias vector of `e(b)`
`e(rmse)`	estimated RMSE vector of `e(b)`
`e(MSE)`	estimated MSE matrix of `e(b)`
`e(ci)`	matrix of confidence intervals
`e(t_ratios)`	$k_2 \times 1$ vector of observed t-ratios in the normal location model
`e(wals_pm)`	$k_2 \times 1$ vector of posterior means in the normal location model
`e(wals_wgt)`	$k_2 \times 1$ vector of WALS weights (diagonal elements of matrix Λ)
`e(priorpar)`	vector of prior parameters

Functions

`e(sample)`	marks estimation sample