earthUI: An Interactive Interface for the earth (MARS) Package

William Bert Craytor

Vol. 1, Issue 1 · Summer 2026
Published July 1, 2026
DOI: 10.67330/7fj41b65

Abstract. earthUI is a graphical user interface for the R `earth` package, which implements Multivariate Adaptive Regression Splines (MARS). It offers three purpose modes—general predictive modeling, real-estate appraisal, and market-area analysis—and guides the user through data import, model configuration, fitting, and the interpretation of earth's diagnostics and graphical output. This article documents earthUI's data-format requirements, modeling workflow, output displays, and complete feature reference.

Keywords: MARS, multivariate adaptive regression splines, earth, R, graphical user interface, real estate valuation

Publisher's Note (July 2026): This article describes earthUI 0.10.0. Shortly before press, earthUI 0.11.0 absorbed the companion applications glmnetUI and mgcvUI (described in the following two articles), consolidating the MARS, elastic-net, and GAM workflows in a single application. The interface and workflow described here are unchanged by the consolidation.

Contents

1 Introduction

1.1 What Is earthUI?

earthUI is a graphical user interface for the R `earth` package, which implements Multivariate Adaptive Regression Splines (MARS) (Friedman 1991). The modelling engine is Stephen

Milborrow's `earth` package (Milborrow 2024), documented in detail by Milborrow (n.d.[a]) and Milborrow (n.d.[b]). It runs as a local Shiny application — there is no login, no server, and no accounts. You launch it from R, import a dataset (CSV or Excel), configure your model, and fit it interactively.

The application provides a complete workflow: data import, variable configuration, model fitting, diagnostic plots, variable importance, model equations, and downloadable reports in HTML, Word, or PDF format.

1.2 One Package, Three Routines

As of the current release, the earthUI package also *contains* its two Post Processing Modules, which in earlier releases were the separate packages glmnetUI and mgcvUI. Installing earthUI installs all three routines; each runs as its own local Shiny application:

- `earthUI::launch()` — the earth (MARS) application described in this article, on port 7878. earth() is the central valuation engine.
- `earthUI::launch_glmnet()` — the elastic net Post Processing Module (port 7879), which imports the earth model's basis functions and re-estimates them under lasso/elastic-net regularization (see "glmnetUI Import" below).
- `earthUI::launch_mgcv()` — the GAM Post Processing Module (port 7880), which uses the earth model's knot locations as starting points for `mgcv` smooth terms (see "mgcvUI Auto-Export" below).

The three routines share one project tree, one settings database, and one copy of the supporting infrastructure, so a project opened in one application is immediately available to the others. The glmnet and mgcv routines are supporting evidence for the earth model's value conclusion, not competing workflows: the intended pattern is to fit and refine the earth model first, then run the modules against the same project when corroboration is wanted.

1.3 Three Purpose Modes

earthUI organizes work into **projects** — named entities that hold your input files, outputs, and settings together at a fixed location. Every project carries a **Purpose**, chosen in the New Project wizard when the project is created. There are three modes:

- **General** — Earth regression for any type of population or dataset. This is the default mode. It provides the full MARS modeling workflow without any domain-specific additions.
- **For Appraisal** — Earth regression tailored for real estate appraisal. Adds features specific to single-property valuation, including subject property handling, special column designations, Residual Constraint Approach (RCA), and Sales Comparison Grid generation.
- **Market Area Analysis** — Earth regression tailored for market area studies. Adds features for analyzing groups of properties in a defined market rather than valuing a single subject.

In all three modes, the core modeling engine is identical — you are always fitting an Earth (MARS) model. The purpose setting controls which additional tools and interface elements are available.

Purpose is fixed per project. A project's purpose — along with its country, state, county, and city — is locked when the project is created. Switching to a different project resets earthUI to a clean default state (imported data, model results, tabs, variable configuration, and earth parameters are all cleared) and then restores that project's saved configuration for its purpose.

1.4 Real Estate–Specific Features

When either **For Appraisal** or **Market Area Analysis** is selected, earthUI activates several features designed for real estate analysis:

- **Special column designations** — Each predictor can be tagged with a special role such as `contract_date`, `dom`, `concessions`, `latitude`, `longitude`, `living_area`, `lot_size`, `actual_age`, `effective_age`, `area`, `site_dimensions`, or `display_only`. These designations control how the column is handled during fitting, output, and Sales Grid generation. See Chapter 6 for the complete reference.
- **Rounding of latitude and longitude** — Columns designated as latitude or longitude are automatically rounded to 3 decimal places to prevent overfitting.
- **Sale Age column** — When a column is designated as `contract_date` and an Effective Date is provided, earthUI computes a `sale_age` column (days between sale date and effective date) and substitutes it as a predictor.
- **RCA computations (appraisal only)** — In appraisal mode, after fitting the model earthUI can compute Residual Constraint Approach (RCA) output, which produces per-comparable adjustments, net/gross adjustment summaries, and an adjusted sale price for the subject property.
- **Sales Comparison Grid (appraisal only)** — Generates a multi-sheet Excel workbook with formulas for comparable adjustments, residual feature breakdowns, and adjusted sale prices. See Chapter 11.

> ***Tip:*** *The General purpose mode works for any dataset — financial, scientific, engineering, or real estate. The appraisal and market modes simply add convenience features for real estate professionals.*

1.5 Installing earthUI and vProlog

earthUI is published on CRAN, so the standard installation is one line in R:

```
install.packages("earthUI")
```

This article describes the current release. If CRAN is still carrying an older version when you read this, the development version installs from GitHub (note the `subdir` — the package lives in the repository's `pkg/` folder):

```
remotes::install_github("wcraytor/earthUI", subdir = "pkg")
```

vProlog (optional). The experimental remarks-processing facility (see "Prolog Text Extraction" near the end of this article) uses the vProlog package — a self-contained embedded Prolog engine. It is deliberately *not* a hard dependency: earthUI installs and runs without it, and the Prolog features simply stay disabled until it is present. vProlog is distributed through the author's r-universe repository, which serves prebuilt binaries for Windows and macOS (no compiler toolchain needed):

```
install.packages("vProlog",
                 repos = c("https://wcraytor.r-universe.dev",
                           getOption("repos")))
```

No system Prolog installation (SWI-Prolog or otherwise) is required — the Trealla engine is compiled into the package itself. Everything else in this article works without vProlog.

1.6 Getting Started

To use earthUI:

1. **Install the package** from CRAN as described above.
2. **Launch the application**: run `earthUI::launch()` in R, or from the command line run `Rscript -e 'earthUI::launch()'`. The app opens in your web browser on port 7878. You can also access the app directly by navigating to `http://localhost:7878` in your browser. (The Post Processing Modules start the same way: `launch_glmnet()` on port 7879, `launch_mgcv()` on port 7880.)
3. **Create or open a project** in Section 1 of the sidebar. The New Project wizard captures the project name, Purpose (General, For Appraisal, or Market Area Analysis), location (country, state, county, city), and optionally an initial data file, which is copied into the project's `in/` folder.
4. **Import your data** in Section 2, which lists the CSV and Excel files in the active project's `in/` folder. Files you add manually via Finder or Explorer appear after clicking Refresh.
5. **Configure variables** — choose your target(s) and predictors, set data types, and assign any special column roles.
6. **Set Earth parameters** — degree, nprune, subset filters, and other `earth()` arguments.
7. **Fit the model** — click "Fit Earth Model" and review the results in the main panel.
8. **Export** — download predictions as Excel, generate Quarto report bundles, or (in appraisal mode) compute RCA adjustments and Sales Comparison Grids.

Configuration is saved server-side in a project database, keyed by project and purpose, via the "Save current as default" buttons in Sections 3 and 4 (see Chapter 3, "Settings Persistence").

2 MLS Input Data Requirements

For real estate appraisal and market analysis workflows, your input data typically comes from a Multiple Listing Service (MLS) export. This chapter describes the expected file structure and the columns that earthUI can use.

2.1 File Format & Structure

earthUI accepts **CSV** and **Excel** (`.xlsx`, `.xls`) files. On import, column names are automatically converted to `snake_case` — for example, "Living SqFt" becomes `living_sqft`, "Contract Date" becomes `contract_date`, and "Sale Price" becomes `sale_price`. This normalization ensures consistent column references throughout the workflow. The CSV separator and decimal mark used during import are determined by the locale settings (see Chapter 3, "Locale & Regional Settings").

Your data file should be a flat table with one row per property and one column per attribute. The first row of the file must contain column headers.

2.2 Required Columns for Appraisal Mode

While earthUI works with any set of columns, the full appraisal workflow (RCA adjustments + Sales Comparison Grid) benefits from having the following columns in your MLS export:

Column	Special Type	Purpose
Sale Price	(target)	Response variable for the model
Contract Date	`contract_date`	Used to compute `sale_age` (days from effective date)
Listing Date	`listing_date`	Used with contract date to compute DOM if no DOM column exists
Days on Market	`dom`	Days on market; displayed in Sales Grid
Concessions	`concessions`	Sale concessions; Net SP = Sale Price − Concessions in Sales Grid
Living Area (SF)	`living_area`	Enables per-SF residuals (`residual_sf`, `cqa_sf`)
Lot Size	`lot_size`	Grouped in "Site Size | Dimensions" row in Sales Grid
Site Dimensions	`site_dimensions`	Grouped with lot size in Sales Grid
Latitude	`latitude`	Rounded to 3 dp; used for proximity calculation and Location group
Longitude	`longitude`	Rounded to 3 dp; used for proximity calculation and Location group
Area ID	`area`	Grouped in "Loc: Long | Lat | Area" row in Sales Grid
Actual Age	`actual_age`	Grouped in "Actual Age | Effective Age" row in Sales Grid
Effective Age	`effective_age`	Grouped with actual age in Sales Grid
Parcel Number	(none)	Displayed in APN row of Sales Grid
Listing ID	(none)	Displayed in APN row of Sales Grid
Address	`display_only`	Displayed in Address row of Sales Grid; excluded from model

Spreadsheet column names can be in a foreign language — the "special" names are in English so that the R program can give them special treatment. Otherwise, the given column names show up in the regression models, graphs, and (if doing appraisals) the Intermediate Sales Grid.

Not all columns are required. earthUI adapts — if a column is missing, the corresponding feature is simply omitted. For example, if no `concessions` column is designated, the Net SP

row in the Sales Grid shows Sale Price without a concessions deduction. However, for real estate pricing models certain columns are highly recommended to achieve acceptable fit:

1. **Sale Age** — the number of days between the contract sale date and the effective date of the appraisal or analysis. If multi-year sales history is being used, especially for periods over 5 years, `sale_age` often plays a central role in estimating the sale price. In fact it is often so important that without it, earthUI fails to provide any model at all.
2. **Living Area** — also goes by names such as "Living Sqft," "GLA" (gross living area) and so on. This is another leading determinant of sale price.
3. **Total Bath Count** — the total number of full, quarter, half, and 3/4 bathrooms. For example, two full baths and one half-bath would be a value of 2.5.
4. **Garage Bays** or **Garage Area** — the number of garage spaces or the garage square footage.
5. **Lot Size** — the land area of the property, typically in square feet or acres.
6. **Longitude, Latitude,** and if available **Area ID**. Adjustments for these will be combined under a single Location adjustment in the Sales Grid.

2.3 Special Column Naming Conventions

earthUI identifies columns by their **special type designation**, not by their column name. You can name your columns anything you like in the MLS export — what matters is that you assign the correct special type in the Variable Configuration table (Chapter 6).

For example, your MLS might export living area as "GLA", "Living SqFt", "liv_area", or "gross_living_area". After import (where it becomes snake_case), you simply designate it as `living_area` in the Special dropdown. earthUI will then use it for per-SF residual calculations and Sales Grid grouping regardless of its original name.

2.4 Data Quality & Completeness

- **Missing values (NA)**: Rows with NA values in any predictor or target column are automatically removed before fitting. The `na.action` is always set to `na.fail` internally, with NA removal handled before the call to `earth()`.
- **Date columns**: Dates should be in a format R can parse (e.g., `2025-06-15`, `06/15/2025`, `June 15, 2025`). earthUI auto-detects date columns when at least 80% of values parse successfully. Two-digit years (e.g., `06/15/25`) are read as the nearest matching year, so recent MLS exports parse correctly; four-digit years are always taken literally.
- **Numeric columns**: Sale price, living area, lot size, concessions, and similar fields must be numeric. If your MLS exports prices with currency symbols or thousands separators (e.g., "$350,000" or "350.000,00"), you may need to clean these before import.
- **Factor columns**: Categorical variables like area ID, style, or condition should contain a manageable number of unique values. earthUI auto-detects factors but you can override the detection.

> ***Tip:*** *Review the T/F/NA column in the predictor table after import — it shows true/false/NA counts for each column. Columns with more than 30% missing values are flagged in red. Consider excluding high-NA columns or cleaning the data before import.*

2.5 Subject Row Placement

In **Appraisal** mode, **row 1 must be the subject property**. All remaining rows are comparable sales. The subject's sale price can be left blank (NA) or set to any value — earthUI treats it as NA during fitting regardless.

In **Market Area Analysis** mode, placing a subject in row 1 is optional, and no row is excluded automatically. To hold a subject row (or any other row) out of the fit, give it a zero value in a designated weights column or use a subset filter.

In **General** mode, there is no special row handling — all rows are treated equally.

3 General Purpose Mode

3.1 Overview

General Purpose mode is the default when you launch earthUI. It provides the complete MARS modeling workflow for any dataset — not just real estate. You can use earthUI for scientific data, financial analysis, engineering studies, or any regression problem where you want to explore non-linear relationships and interactions between variables.

In General mode, the interface omits the real estate–specific features (special columns, sale age, coordinate rounding, RCA). The sidebar is streamlined to focus on variable selection, parameter configuration, model fitting, and export.

3.2 The Sidebar Workflow

The sidebar is organized into numbered, collapsible sections that guide you from project selection through export:

1. Project — Pick an existing project or create a new one via the New Project wizard, which captures the purpose, country, and administrative location (all locked once created). The active project determines where input files are read from and where all outputs are written — there is no separate output-folder field.

2. Import Data — Choose a CSV or Excel file from the active project's `in/` folder; a Refresh link rescans the folder for files added manually. For Excel files with multiple sheets, a sheet selector appears. Column names are automatically converted to snake_case.

3. Variable Configuration — Target variable selector (supports multiple targets), Effective Date, and the predictor table with checkboxes for Include, Factor, Linear, and Latent. See Chapter 6 for full details.

4. Earth Call Parameters — All arguments to the `earth()` function: degree, penalty, nk, pruning method, cross-validation, and more. See Chapter 7 for the complete parameter reference.

Execute Prolog Processing — An optional, unnumbered section between Sections 4 and 5, disabled unless the experimental Prolog feature is enabled in Settings. See Chapter 14.

5. Fit Earth Model — A random-seed field and the button that runs the model asynchronously. See Chapter 8 for fitting details.

6. Download Estimated Target Variable(s) & Residuals — (titled "Download Estimated Sale Prices & Residuals" in appraisal and market modes) Exports predictions, residuals, CQA scores, and per-g-function contributions as an Excel file. See Chapter 9.

7–8. RCA Adjustments and Sales Grid — Appraisal mode only. See Chapters 10 and 11.

9. Generate Quarto Report — Writes a self-contained Quarto report bundle (`.qmd` source plus assets) to the project's output folder. Numbered 7 outside appraisal mode. See Chapter 12.

10. Convert Quarto Report — Renders any `.qmd` file to HTML, Word, or PDF. See Chapter 12.

3.3 Main Panel Tabs

After fitting, the main panel provides ten tabs:

Tab	Contents
Data	Preview of imported data. In appraisal mode, split into Subject Property and Comparable Sales tables.
Equation	The fitted model equation displayed in LaTeX/MathJax, showing each basis function and its coefficient.
Summary	Key metrics (R^2, GRSq, GCV, RSS, CV R^2) and a coefficients table.
Variable Importance	Bar chart and ranked table of predictor importance scores.
Contribution	g-function contribution plots. Select a g-function from the dropdown; univariate predictors show line plots, bivariate interactions (degree ≥ 2) show 3D perspective and contour plots.
Correlation	Heatmap of predictor correlations.
Diagnostics	Three plots: Residuals vs Fitted, Normal Q-Q, and Actual vs Predicted.
RCA Adjustments	(appraisal only) Histograms of Residual Adj %, Net Adj %, and Gross Adj % across the comparables, populated after Step 7 (Calculate RCA Adjustments) completes.
Residual Contribution	Estimates market contributions of rare or excluded features by jointly regressing model residuals on columns the model never used (see subsection 4.4).
ANOVA	ANOVA decomposition table showing each basis function's contribution to RSS.
Earth Output	Raw text output from 'earth::summary()', including the pruning pass details.

3.4 Settings Persistence

earthUI saves your configuration server-side in a project database (`projects.sqlite` in the regProj root), keyed by project and purpose. Saving is **button-driven**: the "Save current as default" button in Section 3 saves the target, effective date, and predictor configuration, while the button in Section 4 saves the earth call parameters and the Allowed Interactions matrix. Each button updates only its own half, so they never clobber each other, and **fitting does not auto-save** — you can experiment freely without overwriting your saved defaults. A project's several input files (a small test extract, the full dataset) share one configuration, and because settings live in the project tree, they travel with it when you sync, back up, or share the regProj folder.

3.5 Dark Mode

Click the moon/sun icon in the upper-right corner to toggle between light and dark themes. The theme preference is saved in local storage and persists across sessions.

3.6 Locale & Regional Settings

earthUI supports international number, date, and CSV formatting conventions through a country-based locale system. The **Country** dropdown in the **Settings** (gear) menu of the top menu bar selects a preset for 31 supported countries. Each preset configures:

- **CSV separator** — comma (`,`) for US/UK/Japan or semicolon (`;`) for most of Europe, where the comma is used as a decimal mark.
- **Decimal mark** — period (`.`) or comma (`,`).
- **Thousands separator** — comma (US/UK/Japan), period (Germany/Italy/Spain), space (Finland/France/Poland/Baltics/Ukraine/Russia), or apostrophe (Switzerland).

- **Date format** — MM/DD/YYYY (US), DD/MM/YYYY (most of Europe), or YYYY-MM-DD (Sweden/Lithuania/Japan/Canada).
- **Paper size** — Letter (US/Canada/Mexico) or A4 (everywhere else).

3.6.1 Supported Countries

Country	CSV Sep	Decimal	Thousands	Date	Paper
US, Canada (csv)	,	.	,	MDY/YMD	Letter
UK, Ireland, AU, NZ	,	.	,	DMY	A4
Germany, Austria, Belgium, Netherlands, Italy, Spain, Portugal, Denmark, Brazil	;	,	.	DMY	A4
France, Poland, Czech Rep., Norway, Latvia, Estonia	;	,	(space)	DMY	A4
Finland, Sweden	;	,	(space)	DMY/YMD	A4
Switzerland	;	,	’	DMY	A4
Ukraine, Russia	;	,	(space)	DMY	A4
Turkey	;	,	.	DMY	A4
Japan, South Korea	,	.	,	YMD	A4
Mexico	,	.	,	DMY	Letter
Lithuania	;	,	(space)	YMD	A4

3.6.2 Paper Override

Below the country selector, a **Paper** dropdown (Letter or A4) lets you override the report page size without switching countries. The CSV separator, decimal mark, and date-format order follow the selected country’s preset; when you change the country, the paper override resets to that country’s default.

3.6.3 Saving Defaults

Click **Save** in the Settings menu to store your locale preferences globally. These defaults apply to all future sessions regardless of which project you open. Per-project settings (target, predictors, parameters) are saved separately in the project database; locale defaults persist across all projects via a user-level SQLite database. This two-level approach is designed for organizations like audit firms that work with data from multiple countries — set your most common country as the default, then adjust when needed.

> ***Tip:*** *Number formatting on plot axes and slope labels automatically adapts to your locale. German locale uses periods for thousands (200.000), Finnish uses spaces (200 000), Swiss uses apostrophes (200’000). No currency symbols are displayed — earthUI is currency-agnostic.*

4 Appraisal Mode

When you select **For Appraisal** as the Purpose, earthUI configures itself for single-property valuation. All features described in Chapter 3 remain available; this chapter covers only the appraisal-specific additions.

4.1 Subject Row Handling

In appraisal mode, **row 1 of your dataset is the subject property** and all remaining rows are comparable sales. Your input file must be organized accordingly (see Chapter 2). The subject's sale price can be left blank or set to any value — earthUI automatically treats it as NA during fitting.

After importing, the Data tab splits into two sections: **Subject Property** (row 1) and **Comparable Sales** (rows 2+). Row 1 is always excluded from model fitting — the notification "Skipping row 1 (subject). Fitting on N rows." confirms this. After fitting, the model still generates predictions for the subject row, shown as `est_<target>` in the output.

4.2 Effective Date & Sale Age

An **Effective Date** field appears in the Variable Configuration section in every purpose mode (defaulting to today's date) — general and market analyses have an effective date too, not just appraisals — but it is only *used* in appraisal and market modes. There, if you designate a column as `contract_date` in the Special column dropdown, earthUI computes a `sale_age` column — the number of integer days between each sale's contract date and the effective date. This column replaces the original date column as a predictor.

The first time you click Fit after designating a contract date, earthUI creates the `sale_age` column and notifies you to click Fit again to include it.

4.3 Special Column Designations

In appraisal and market modes, a **Special** dropdown appears for each predictor in the Variable Configuration table. The complete list of special types and their effects:

Special Type	Effect
`no`	Default — no special handling
`contract_date`	Triggers automatic `sale_age` computation from the Effective Date
`listing_date`	Used with contract date to compute DOM when no `dom` column exists
`dom`	Identifies the Days on Market column; displayed in Sales Grid
`concessions`	Identifies sale concessions; used in Net SP formula (Sale Price − Concessions) in Sales Grid
`latitude`	Values rounded to 3 dp; used for Haversine proximity and Location group in Sales Grid
`longitude`	Values rounded to 3 dp; used for Haversine proximity and Location group in Sales Grid
`area`	Grouped in "Loc: Long | Lat | Area" row in Sales Grid
`living_area`	Enables per-SF residuals (`residual_sf`, `cqa_sf`) in download output
`lot_size`	Grouped in "Site Size | Dimensions" row in Sales Grid
`site_dimensions`	Grouped with lot size in Sales Grid
`actual_age`	Grouped in "Actual Age | Effective Age" row in Sales Grid
`effective_age`	Grouped with actual age in Sales Grid
`display_only`	Column included in exports but excluded from model fitting (multiple allowed)
`remarks`	(Experimental) Free-text remarks column parsed into feature columns by the optional Prolog step (multiple allowed); see Chapter 14
`list`	(Experimental) Comma-delimited column split into one-hot TRUE/FALSE columns by the optional Prolog step (multiple allowed); see Chapter 14

Only one column per special type is allowed (except `display_only`, `remarks`, and `list`). Assigning a special to a second column automatically clears it from the first. A small blue badge appears next to the variable name showing its assigned special type.

4.4 Residual Contribution (Rare-Feature Estimation)

Features too rare for the regression channel — a workshop present in a handful of sales — belong to the RCA residual layer. The **Residual Contribution** tab estimates their market contribution by regressing the fitted model's residuals on columns the model never used, via ordinary least squares. Selecting several columns estimates them *jointly*, so correlated amenities (e.g. `outbuilding_sqft` plus `pr_has_workshop`) split the credit instead of double-counting. Binary flags yield lump-sum estimates and list each flagged sale's residual as evidence; numeric columns yield per-unit contributions; the reported standard error is one SE ($t \geq 2$ as the usual significance yardstick). Terms supported by fewer than ten observations receive an automatic CAUTION in the generated workfile note, which documents method, estimates, and caveats in one paragraph for the report. Only predictors the model actually *used* are excluded from the candidate list — a rare flag offered to the fit but never selected by the forward pass remains eligible, since the model absorbed none of its value.

4.5 RCA Adjustments Overview

The **Calculate RCA Adjustments & Download** button (sidebar section 7, visible only in appraisal mode after fitting) computes market-derived adjustments for each comparable relative to the subject. The full RCA workflow is described in Chapter 10.

After computing RCA adjustments, the **Generate Sales Grid & Download** button (sidebar section 8) becomes available. The Sales Grid workflow is described in Chapter 11.

Tip: *The CQA score you assign to the subject controls how much of the residual distribution is attributed to the subject. A score of 5.00 places the subject at the median of the comparables.*

5 Market Area Analysis Mode

When you select **Market Area Analysis** as the Purpose, earthUI provides the same real estate–specific features as appraisal mode (special columns, sale age, coordinate rounding) but is oriented toward analyzing a group of properties rather than valuing a single subject.

5.1 Differences from Appraisal Mode

- **No automatic subject exclusion** — unlike appraisal mode, market mode does not automatically exclude row 1 from fitting. All rows are included; to hold a subject row out of the fit, give it a zero value in a designated weights column (as in the demo dataset) or use a subset filter.
- **No RCA or Sales Grid sections** — the "Calculate RCA Adjustments & Download" and "Generate Sales Grid & Download" steps are not available. Market mode focuses on model fitting and output, not per-comparable adjustments.
- **Sidebar numbering** — without the RCA and Sales Grid steps, the Generate Quarto Report section is numbered 7 instead of 9 (the Convert Quarto Report section remains 10).

5.2 When to Use Market Mode

Market Area Analysis mode is appropriate when you are:

- Building a regression model for a neighborhood or market area to understand value drivers
- Analyzing how variables like square footage, age, lot size, and location affect sale prices across a group of properties
- Preparing support for a market conditions analysis or neighborhood delineation
- Working with a dataset that includes a subject property in row 1 but you want the option of including or excluding it from the fit

> ***Tip:*** *Market mode is also useful for general real estate regression where you want special column features (sale age, coordinate rounding) but do not need the RCA adjustment workflow.*

6 Variable Selection

Section 3 of the sidebar — **Variable Configuration** — is where you choose which columns participate in the model and how they are treated.

6.1 Target Variable(s)

The **Target (response) variable(s)** dropdown at the top of Section 3 lists every column in your dataset. Select one column for a standard single-response model, or multiple columns for a multi-response model. When multiple targets are selected, the model fits all responses simultaneously using `earth(cbind(y1, y2, ...) ~ .)`.

Columns selected as targets are automatically excluded from the predictor list.

6.2 The Predictor Table

Below the target selector, a table lists every remaining column with the following fields:

Column	Description
Variable	Column name (full name shown in tooltip if truncated), preceded by a ×/+ **toggle** (see below). In appraisal/market modes, a blue badge shows the assigned special type.
Type	Data type dropdown: `numeric`, `integer`, `character`, `logical`, `factor`, `Date`, `POSIXct`
Include	Checkbox — include this column as a predictor in the model
Factor	Checkbox — treat this column as a categorical variable
Linear	Checkbox — force linear entry only (no hinge functions)
Latent	Checkbox — hold this column out of the model entirely (excluded from earth and from the data exported to glmnetUI / mgcvUI), reserving it for latent-variable work
Special	Dropdown (appraisal/market only) — see Special Column Types Reference below
T/F/NA	Three counts per column: for logicals, TRUE/FALSE/NA; for numerics, $\neq 0$/$= 0$/NA. Shown in red when more than 30% of values are missing

A hint line above the table explains the abbreviations: "Type = column data type, Inc = include as predictor, Factor = treat as categorical, Linear = linear-only (no hinges), Special = column role (e.g. contract_date)."

Soft-disable instead of delete. The × toggle beside each variable name greys out the row and holds the column out of the model without removing it from the data; it flips to a green + that re-enables the column. Columns are never destroyed.

6.3 Data Type Detection & Overrides

earthUI automatically detects data types on import. Numeric, integer, logical, factor, and date columns are recognized. Character columns that look like dates (at least 80% of values parse against common date formats) are classified as `Date`.

You can override any detection by changing the **Type** dropdown. When you change a column to `character` or `factor`, the Factor checkbox is automatically checked. Changing types affects how the column is passed to the `earth()` function.

6.4 Factor and Linear Flags

- **Factor** — When checked, the column is treated as a categorical variable. This is appropriate for columns like `style`, `area_id`, or `grade` that represent discrete groups rather than continuous measurements. Factor columns enter the model as indicator (dummy) variables.
- **Linear** — When checked, the column is forced to enter the model linearly — no hinge (piecewise-linear) functions are created for it. This is useful when you know a variable has a strictly linear relationship with the target.

6.5 Special Column Types Reference

In appraisal and market modes, the **Special** dropdown provides the following options. Each type can be assigned to at most one column (except `display_only`, `remarks`, and `list`, which allow multiple):

Date & Time Types:

- `contract_date` — Triggers automatic `sale_age` computation. The original date column is replaced by an integer column measuring days between the sale date and the Effective Date.
- `listing_date` — Used as a fallback for computing Days on Market (DOM = contract date − listing date) when no explicit `dom` column is designated.
- `dom` — Identifies the Days on Market column. Displayed in the Sales Grid's APN row and Date of Sale row.

Monetary Types:

- `concessions` — Identifies sale concessions (seller credits, buyer incentives, etc.). Used in the Sales Grid to compute Net Sale Price: Net SP = Sale Price − Concessions.

Size & Location Types:

- `latitude` — Values are automatically rounded to 3 decimal places to prevent overfitting. Used for Haversine proximity calculations (distance from subject to each comp) and grouped in the Location row of the Sales Grid.
- `longitude` — Same rounding treatment as latitude. Used for proximity and the Location group.
- `area` — Typically a neighborhood or area identifier. Grouped with latitude and longitude in the "Loc: Long | Lat | Area" row of the Sales Grid.
- `living_area` — Enables per-square-foot residual calculations (`residual_sf` and `cqa_sf`) in the download output.
- `lot_size` — Grouped in the "Site Size | Dimensions" row of the Sales Grid.
- `site_dimensions` — Grouped with lot size in the Sales Grid (e.g., "75x120").

Age Types:

- `actual_age` — Grouped in the "Actual Age | Effective Age" row of the Sales Grid.
- `effective_age` — Grouped with actual age in the Sales Grid.

Display Types:

- `display_only` — The column is included in Excel exports but excluded from model fitting entirely. Use this for address fields, MLS numbers, parcel IDs, or other reference data that should not be a predictor. Multiple columns can have this designation.

Text Extraction Types (experimental):

- `remarks` — A free-text column (e.g. `public_remarks`) to be parsed into feature columns by the optional Prolog step. Does nothing until that feature is enabled in Settings. Multiple columns allowed. See Chapter 14.

- `list` — A comma-delimited column (e.g. `existing_structures`) to be split into one-hot `<col>_<value>` TRUE/FALSE columns by the same step. Multiple columns allowed. See Chapter 14.

6.6 Multiple Targets

Selecting more than one target variable fits a multi-response Earth model. When multiple targets are selected:

- The model predicts all responses simultaneously, sharing basis functions across targets
- The **wp (response weights)** button becomes active, allowing you to assign a numeric weight to each target
- Variance models (`varmod.method`) are disabled (not supported for multi-response)
- Results tabs display per-response metrics, equations, and diagnostic plots

> ***Tip:*** *Columns designated as* ***`display_only`*** *remain in your dataset and appear in the Excel export, but are excluded from the predictor table entirely. Use this for ID columns, addresses, or other reference fields.*

7 Parameter Selection

Section 4 of the sidebar — **Earth Call Parameters** — provides access to all arguments accepted by the `earth()` function. Each parameter has a blue help icon (**?**) with a tooltip explanation. Parameters are organized into collapsible subsections.

7.1 Basic Parameters

Parameter	Default	Description
subset	(empty)	Row filter expression. See "Subset Filtering" below.
weights	NULL	Column selector for case (row) weights. Only numeric columns are listed.
wp	NULL	Response weights for multi-target models. Button opens a dialog with one numeric input per target (default 1.0 each).
keepxy	off	Retain x, y, subset, and weights in the model object.
trace	0	Trace level for fitting output (0 through 5).
glm	none	Optional GLM family: `gaussian`, `binomial`, or `poisson`.
degree	1	Maximum interaction order. Setting degree ≥ 2 auto-enables cross-validation and reveals the Allowed Interactions matrix.
penalty	2	GCV penalty per knot. Higher values produce simpler models.

7.2 Forward Pass

Parameter	Default	Description
nk	auto	Maximum terms before pruning. Auto-filled with the recommended value for the loaded data (see "Recommended Parameter Values" below).
thresh	0.001	Forward-step threshold. Smaller values allow more terms.
minspan	auto	Minimum span between knots. Auto-filled with the recommended value. Negative values set the maximum knots per predictor.

Parameter	Default	Description
endspan	auto	End span — minimum distance from a knot to the edge of the data. Auto-filled with the recommended value.
newvar.penalty	0	Penalty for introducing a new variable (encourages reuse of existing predictors).
fast.k	20	Number of parent terms to consider in the fast MARS algorithm.
fast.beta	1	Controls the fast MARS aging factor.

7.3 Allowed Interactions

When **degree** is set to 2 or higher, an interaction matrix appears below the basic parameters. This is an upper-triangular grid of checkboxes, one for each predictor pair. A checked box means the two predictors are allowed to interact; unchecking it forbids that specific interaction.

Clicking a predictor name (row or column label) toggles all interactions for that variable. **Allow All** and **Clear All** checkboxes at the top provide bulk control. The matrix uses sticky headers so column and row labels remain visible when scrolling.

An info alert reminds you: "Interaction terms increase the risk of overfitting. Cross-validation has been enabled (10-fold)."

7.4 Pruning

Parameter	Default	Description
pmethod	backward	Pruning method: `backward`, `none`, `exhaustive`, `forward`, `seqrep`, or `cv`.
nprune	NULL	Maximum terms after pruning. Leave empty for automatic selection.

7.5 Cross-Validation

Parameter	Default	Description
nfold	10	Number of cross-validation folds. Set to 0 to disable CV.
ncross	20	Number of cross-validation repetitions.
stratify	on	Stratify CV samples so each fold has a similar response distribution.

When degree ≥ 2, cross-validation is automatically enabled (nfold set to 10) to help guard against overfitting from interactions.

7.6 Subset Filtering

The **subset** text input accepts an R expression that filters which rows are used for model fitting. You can type an expression directly (e.g., `sale_age < 365 & area_id == 460`) or use the **Build filter...** button to construct one visually.

Build Filter Dialog — Click "Build filter..." to open a guided dialog. Each condition row has a column dropdown, operator (`<`, `>`, `<=`, `>=`, `==`, `!=`), and a value input that adapts to the column type: numeric input for numbers, date picker for dates, and dropdown of unique values for character/factor columns. Conditions are joined with AND (`&`) or OR (`|`) connectors. A preview at the bottom shows the expression and how many rows match. Click **Apply** to insert the expression into the text input.

Date columns must use `as.Date("...")` or `as.POSIXct("...")` wrappers in manual expressions. The Build Filter dialog handles this automatically.

Subset filtering is non-destructive — excluded rows remain in the dataset and receive predictions in the Excel export.

> ***Tip:*** *Use parentheses to group OR conditions when combining with AND. Without parentheses,* `A & B | C` *is evaluated as* `(A & B) | C`*, which may not be what you intend.*

7.7 Recommended Parameter Values

earthUI displays a recommended value below key parameters in Section 4. These recommendations update reactively based on the number of fitting rows (n) and selected predictors (p). The formulas are derived from Friedman's MARS paper, earth's internal algorithms, and empirical testing. The **nk**, **minspan**, and **endspan** fields are pre-filled with their recommended values as the data loads; a manual edit is preserved.

7.7.1 Forward Pass Parameters

nk (max terms before pruning):

$$\text{nk} = \min\big(100,\ \max(21,\ 2p + 1,\ \lfloor n/10 \rfloor)\big)$$

n	p=5	p=10
30	21	21
50	21	21
100	21	21
200	21	21
500	50	50
1000	100	100
1500	100	100

Earth's default, $\min(200, \max(20, 2p)) + 1$, does not account for dataset size and can be too small for large datasets, constraining the forward pass before it explores all predictors adequately. The $\lfloor n/10 \rfloor$ term allows approximately one term per 10 observations — a standard rule of thumb for avoiding overparameterization. The cap of 100 prevents diminishing returns.

minspan (minimum observations between knots):

$$\text{minspan} = \min\big(16,\ \lfloor 5 + n/50 \rfloor\big)$$

endspan (minimum observations from data boundaries):

$$\text{endspan} = \min(16,\ \lfloor 5 + n/28 \rfloor)$$

n	minspan	endspan
30	5	6
50	6	6
100	7	8
200	9	12
300	11	15
500	15	16
1500	16	16

Earth's auto-calculated minspan uses Friedman's equation 43: $\lfloor(-\ln(-\ln 0.95) + \ln(p \cdot n))/(2.5 \ln 2)\rfloor$, which scales as $\ln(np)$ and grows too slowly for large datasets. At n=1500 it yields only ≈7, giving roughly 245 candidate knot locations per continuous predictor — too many, allowing the forward pass to fit noise. The recommended formula targets approximately 100 candidates per predictor for large n, while deferring to earth's auto-calculation for small n (where it is well-tested). Endspan is set slightly larger than minspan to provide additional boundary protection, which helps when data has thin tails (common in real estate).

penalty (GCV penalty per knot):

$$\text{penalty} = \begin{cases} 3 & \text{if degree} > 1 \\ 2 & \text{if degree} = 1 \end{cases}$$

This is earth's own default — 2 for additive models, 3 for interaction models (the higher penalty compensates for the larger search space).

newvar.penalty (penalty for introducing a new predictor):

Recommended: **0.1** when predictors are correlated (e.g., living area with bedroom count, bathroom count, lot size). This biases the forward pass toward reusing predictors already in the model rather than introducing correlated alternatives. The mechanism: each new predictor's RSS improvement is multiplied by $1/(1 + \text{penalty})$. With newvar.penalty = 0.1, a new variable must improve RSS by at least 10% more than another knot on an existing variable. This produces simpler models with fewer predictors without affecting final coefficient estimates (the penalty is removed after selection). Leave at 0 if predictors are not correlated. Note that earth does not support a non-zero `newvar.penalty` together with case weights — earthUI disables the input (forcing 0) whenever a weight column is designated.

7.7.2 Pruning Parameters

pmethod: Recommended: **backward**. The default GCV-based backward pruning is deterministic — the same data always produces the same model, with no dependence on random seeds. This is important for reproducibility, especially in appraisal work. The `cv` method uses cross-validation for pruning which introduces seed dependence.

nprune: Recommended: **leave empty** (NULL). Let GCV select the optimal number of terms. Setting nprune imposes a hard cap that overrides GCV's judgment.

7.7.3 Cross-Validation Parameters

nfold (CV folds): recommended by sample-size band:

n (fitting rows)	nfold
0–299	5
300–399	6
400–499	7
500–599	8
600–699	9
700+	10

More folds train each fold on more data (a less biased estimate) but shrink the held-out test fold. The table graduates up to the standard 10-fold as the sample grows while keeping each test fold large enough (roughly 50+ rows) for a stable per-fold metric, and falls back to 5-fold for small samples — where the CVR^2 estimate is still noisy, so raise ncross (e.g. 15–20) to average out the variance. The cap of 10 reflects that each additional fold is a full earth fit for negligible diagnostic gain. With `pmethod = "backward"`, cross-validation does not affect the model — it only computes the diagnostic CVR^2 and provides residuals for the variance model.

ncross (CV repetitions): recommended by sample-size band:

n (fitting rows)	ncross
0–199	20
200–399	16
400–699	12
700–1199	10
1200–2999	7
3000+	5

Repeats average out fold-partition luck to stabilize CVR^2; larger samples need fewer repeats because a single cross-validation is already more stable. The repeated CV residuals also feed the variance model.

7.7.4 Variance Model

varmod.method: Recommended: **lm** — Milborrow's default: a robust linear variance model for prediction intervals (spread is estimated from noisy residuals, so a simple model generalizes better). If `lm` fails to converge on your data, try `rlm` or `const` (homoscedastic, always converges), or fall back to `earth` (also handles a nonlinear spread). It requires `nfold > 1` and `ncross > 1`.

7.7.5 Earth Output File

After every successful fit, earthUI writes `<filename>_earth_output_<timestamp>.txt` to the output folder. This file contains the model terms, summary statistics (R^2, GRSq, CVRSq), variance model details, and trace log. One file is created per fit, providing a cumulative record for comparing parameter configurations.

7.8 Settings Defaults

A radio button at the top of Section 4 controls which defaults are loaded:

- **Use last settings for input file** — restores the settings you used last time with this particular file
- **Use default settings** — applies your saved custom defaults
- **Earth defaults** — resets all parameters to factory values

Click **Save current as default** at the bottom of Section 4 to store the current earth parameters and Allowed Interactions matrix; the matching button in Section 3 stores the target, effective date, and predictor configuration. Both are saved per project and purpose in the project database.

8 Fitting the Earth Model

8.1 The Fit Button

Section 5 of the sidebar contains the **Fit Earth Model** button together with a **Random seed** field. The seed makes cross-validation fold assignments reproducible — a dropdown recalls the last 10 seeds used, and the seed used is recorded in the Earth Output tab after fitting. Before fitting, earthUI validates your configuration — a target must be selected and at least one predictor must be included. In appraisal/market modes, latitude and longitude columns are rounded, and sale age is computed from the effective date and contract date (if designated).

8.2 Fitting Modal & Trace Output

When you click Fit, a dark modal overlay appears with:

- **Header** — "Fitting Earth Model" with an elapsed timer counting seconds and an **Abort** button that cancels the background fit
- **Trace log** — a scrollable, monospace text area showing real-time output from the `earth()` function: dataset info, forward pass progress, cross-validation folds, and completion time
- **Color coding** — default lines in green, cross-validation lines in yellow, errors in red

earthUI fits models asynchronously using `callr::r_bg()`, which runs the computation in a separate R process. This keeps the application responsive during long-running fits. A 300ms polling observer reads output from the background process and streams it to the trace log.

When fitting completes, a "Done in X.Xs" message appears, a close button (X) is added to the modal, and the backdrop is removed so the result tabs can render on demand. The modal then auto-dismisses once the visible outputs finish rendering — or you can close it earlier.

On success, a green checkmark is appended to the Fit button. If an error occurs, the error message is displayed in the trace log and a notification appears.

8.3 Fit Log

Every fit (success or failure) writes a log file to your output folder:

- **Filename**: `<datafile>_earth_log_<YYYYMMDD_HHMMSS>.txt`
- **Contents**: timestamp, all trace output from the fitting process, and any error messages
- **Location**: the active project's output folder

> ***Tip:*** *If callr is not installed, earthUI falls back to synchronous fitting with a simple progress bar. Install callr for the full trace-output experience.*

9 Downloading Data

After fitting, sidebar section 6 — titled "Download Estimated Target Variable(s) & Residuals" in general mode and "Download Estimated Sale Prices & Residuals" in appraisal and market modes — downloads an Excel file with predictions and diagnostics. This output is used in Step 7 (RCA) to assign a CQA (Condition-Quality-Appeal) rating to the subject property. The output is sorted by `residual_sf` and `cqa_sf` to help you assess where the subject falls in the ranking. If the model is good quality, then the properties should be ranked from lowest appealing to most appealing based on residual features that did not go into the regression. The middle value should be approximately 0, the lower half negative values and the upper half positive values. You should find the worst quality homes, or "fixers" near the bottom of the ranking and the nicest homes at the top. There will usually be exceptions for anomalies such as foreclosures, short sales, probate (inheritance related) sales, and quick sales needed for job change or other reasons. Investigation of anomalies usually turns up a pertinent reason for the price anomaly.

The button label adapts to the purpose mode:

- General/Market: **Download Output (Excel)**
- Appraisal: **Download Intermediate Output (Excel)**

The filename format is `<datafile>_modified_<YYYYMMDD_HHMMSS>.xlsx`.

9.1 Output Columns

For each target variable, the following columns are appended:

Column	Description
`est_<target>`	Model prediction, e.g. `est_sale_price` (1 decimal place)
`residual`	Actual minus predicted (1 dp)
`cqa`	Condition-Quality-Appeal score (2 dp, range 0–10)
`residual_sf`	Residual divided by living area (if designated, 1 dp)
`cqa_sf`	CQA calculated from ranking via residual_sf (2 dp)
`<variable>_contribution`	Per-g-function contribution value (1 dp, one column per g-function)
`basis`	Intercept value contribution, same for all properties (1 dp)
`calc_residual`	Actual minus (basis + all contributions) — verification column (1 dp)

For multi-target models, a `_<i>` suffix is added to distinguish columns for each target.

9.2 Column Ordering & Excel Formatting

The ranking columns are placed at the **leftmost position** in the output file, in this order: `residual_sf`, `cqa_sf`, `residual`, `cqa`. This makes it easy to scan the sorted list and evaluate where the subject falls in the CQA distribution.

These columns have Excel number formatting applied:

Column	Format	Example
`residual_sf`	Numeric, 2 decimal places	12.34
`cqa_sf`	Numeric, 2 decimal places	7.25
`residual`	Numeric, 0 decimal places	15,200
`cqa`	Numeric, 2 decimal places	6.80

9.3 CQA Scores

The CQA (Condition-Quality-Appeal) score ranks each row's residual against all other residuals. For a given row, the CQA is the percentage of rows with a smaller signed residual, multiplied by 10. This produces a 0–10 scale where:

- **High CQA** ($\approx$ 9–10) — the property sold for much more than the model predicted (large positive residual)
- **Low CQA** ($\approx$ 0–1) — the property sold for much less than predicted (large negative residual)
- **CQA $\approx$ 5** — the residual is near the median of all residuals

When a `living_area` column is designated, `cqa_sf` provides the same ranking based on per-square-foot residuals.

9.4 Sorting & Subject Row

In appraisal and market modes, comparable rows are sorted by `residual_sf` descending (or `residual` if no living area is designated). The subject row (row 1) remains in position 1 and is not sorted. In general mode, rows are exported in their original order.

This sorting, combined with the leftmost ranking columns, allows the appraiser to quickly scan the comparables from most over-predicted to most under-predicted, and assess where the subject's assigned CQA score falls in that distribution.

Factor levels in the prediction data are aligned with the training data before calling `predict()`. Rows with unseen factor levels will produce NA predictions.

> ***Tip:*** *The `calc_residual` column should always equal `residual` (within rounding). If they differ significantly, it indicates a computation issue worth investigating.*

9.5 mgcvUI Auto-Export

On every successful fit with `degree` $\leq$ 2, earthUI automatically saves the full result object as an `.rds` file to the project's output folder. The filename follows the pattern `<datafile>_earthUI_result_<YYYYMMDD_HHMMSS>.rds`.

This file can be loaded by the mgcv Post Processing Module — included in the earthUI package and started with `earthUI::launch_mgcv()` — using `readRDS()`. The GAM routine uses the earth model's knot locations and basis functions as starting points for GAM smooth terms, enabling a seamless transition from MARS to GAM modeling.

Models with `degree` $>$ 2 are skipped because mgcvUI only supports pairwise interactions. When earthUI is launched in Trilogy mode (from the Trilogy UI), an additional **Lock Model Output** section lets you lock a particular fit's `.rds` — together with its predictor basis and shared CQA settings — as the canonical earth model that glmnetUI and mgcvUI import.

9.6 glmnetUI Import

The same `.rds` file saved for the GAM routine can also be imported into the **elastic net Post Processing Module** — likewise included in the earthUI package and started with `earthUI::launch_glmnet()`. In the glmnet application, navigate to **Section 2: Import from earthUI** and use the Browse button to select the `.rds` file.

glmnetUI uses the standard `earth::model.matrix()` approach: the earth model's basis functions (hinges, interactions, and linear terms) become the columns of glmnet's design matrix. glmnet then performs regularized regression on these basis columns, selecting and shrinking them via lasso/elastic net. This combines earth's adaptive basis construction with glmnet's regularization.

9.6.1 Key Benefits

- **Interpretable interactions**: earth's hinge-based interactions (including 3-way terms) have clear geometric meaning, unlike glmnet's native cross-product interactions which are difficult to explain in court or audit settings.
- **Automatic nonlinearity**: earth's hinge functions capture nonlinear relationships that a standard glmnet model with raw predictors would miss.
- **Variable selection**: glmnet can further prune the earth basis, dropping hinge terms that don't contribute.

9.6.2 Important: Weight Column Must Match

Warning

If earthUI uses a weight column (e.g., to exclude the subject row with weight=0), the same weight column must be designated in glmnetUI's variable configuration. If the weight settings differ, the row counts will not align and the earth basis cannot be appended to glmnet's model matrix.

9.6.3 Workflow

1. Fit an earth model in earthUI (any degree).
2. The `.rds` result file is saved automatically to the project's output folder.
3. In glmnetUI, open **Section 2: Import from earthUI** and browse to the `.rds` file.
4. Verify that the **same data file** and **same weight column** are used in both apps.
5. Configure glmnet parameters in Section 5 (the Interaction Matrix is automatically disabled when an earthUI import is active).
6. Click **Fit Model**. The earth basis columns appear in the Equation, Coefficients, and Contributions tabs.

10 RCA Calculations & Downloading

The **Calculate RCA Adjustments & Download** button appears in sidebar section 7, visible only in appraisal mode after a model has been fitted. RCA (Residual Constraint Approach) produces market-derived, per-comparable adjustments relative to the subject property.

10.1 Opening the RCA Dialog

Clicking the button opens a small modal dialog with:

- **Score type** — radio buttons to choose **CQA** or **CQA per SF** (the SF option is available only when a `living_area` column is designated). If you choose "CQA per SF," then based on the CQA score you assign the subject, its residual score will be its living area times the residual_sf that matches the given CQA_SF score.
- **CQA value** — input for the subject's CQA score (0.00–10.00). This score represents where you believe the subject falls in the quality distribution of the comparables.
- **Generate** button — computes the RCA output and downloads it as Excel. The button stays disabled until a valid score is entered.

10.2 CQA Score Interpolation

When you click Generate, earthUI interpolates the subject's residual from the comparable CQA/residual pairs:

1. The comparables' CQA scores and residuals are sorted by CQA ascending
2. Linear interpolation (`stats::approx()`) maps your entered CQA value to a residual
3. If using CQA per SF, the per-SF residual is converted back to a total residual by multiplying by the subject's living area
4. The subject's estimated value is computed as: model prediction + interpolated residual

10.3 Output Columns

The RCA Excel file (`<datafile>_adjusted_<YYYYMMDD_HHMMSS>.xlsx`) includes all intermediate output columns plus:

Column	Description
`subject_value`	Model prediction + interpolated residual (row 1 only)
`subject_cqa`	The CQA score you entered (row 1 only)
`<variable>_adjustment`	Subject contribution minus comp contribution (per g-function)
`residual_adjustment`	Subject residual minus comp residual
`net_adjustments`	Sum of all adjustments (contribution + residual)
`gross_adjustments`	Sum of absolute values of all adjustments
`residual_pct`	Residual adjustment as a fraction of the comp sale price
`net_adj_pct`	Net adjustments as a fraction of the comp sale price
`gross_adj_pct`	Gross adjustments as a fraction of the comp sale price
`adjusted_sale_price`	Comp sale price + net adjustments

The adjustment columns tell you, for each comparable, how much the model attributes the difference to each variable. `adjusted_sale_price` is the comparable's sale price after applying all model-derived adjustments — a set of adjusted sale prices that should cluster around the subject's estimated value.

10.4 The RCA Adjustments Tab

Once the RCA output has been computed, the **RCA Adjustments** tab in the main panel displays histograms of the Residual Adj %, Net Adj %, and Gross Adj % distributions across the comparables (one set per target for multi-target models), giving a quick visual check on how tightly the adjustments cluster.

> ***Tip:*** *A CQA score of 5.00 places the subject at the median of the comparables. Scores above 5 indicate the subject is above average quality; below 5 indicates below average.*

11 Sales Comparison Grid

11.1 Overview & Purpose

The **Sales Comparison Grid** is an Excel workbook generated in appraisal mode (sidebar section 8) after computing RCA adjustments. It presents the subject property alongside selected comparables in a structured grid format, with Excel formulas that automatically compute adjustments and an adjusted sale price for each comparable.

The grid is designed for the appraiser's workfile — it combines the regression-derived adjustments from the Earth model with editable cells where the appraiser can allocate the CQA residual to specific property features. The output filename is `SalesGrid_<YYYYMMDD_HHMMSS>.xlsx`.

11.2 Comp Selection Modal

Clicking "Generate Sales Grid & Download" opens a modal dialog where you select which comparables to include:

- **Recommended comps** are pre-checked. These are comparables with a gross adjustment percentage below 25% of sale price, sorted by sale age ascending (most recent sales first) and capped at 30.
- **Additional comps** are listed below, unchecked by default. These have gross adjustments of 25% or more but are still available for selection (up to 50 shown, sorted by gross adjustment percentage ascending).
- A maximum of **30 comps** can be selected, producing up to **10 sheets** (3 comps per sheet).

After confirming your selection, the selected comps are sorted by gross adjustment percentage ascending, so the most similar comparables appear on Sheet 1.

11.3 Grid Layout

Each sheet has a 20-column layout accommodating the subject property plus 3 comparables. The columns for each entity are:

Entity	Columns (5 per entity)
Subject	Label, Factual Value 1, Factual Value 2, Factual Value 3, Value Contribution (VC)
Each Comp	Factual Value 1, Factual Value 2, Factual Value 3, Value Contribution (VC), Adjustment

The rows from top to bottom are:

1. **Title** — sheet name (e.g., "Intermediate Sales Comparable Grid — Sheet 1 of 3")
2. **Headers** — "Subject" and comp column headers with address
3. **Address** — full property address
4. **APN | MLS# | DOM | Subj.Prox** — parcel number, listing ID, days on market, and Haversine distance (miles) from subject
5. **Sales Price | Concess. | Net SP** — sale price, concessions, and Net Sale Price formula
6. **Regression Features** header row
7. **BASE VALUE** — the model intercept
8. **Date of Sale | OffMkt | OnMkt** — contract date, sale age, and DOM
9. **Grouped rows** (conditional) — Location, Site Size, and/or Age groups
10. **Model variable rows** — one row per predictor (excluding grouped variables)

11. **Blank separator row**
12. **Residual Features** header row
13. **CQA|Residual** — CQA score + remaining residual formula
14. **Residual feature rows** — named + blank rows for appraiser entry
15. **Total VC / Net Adjustment**
16. **Net Adjustment %**
17. **Gross Adjustment %**
18. **Adjusted Sale Price** — formula row
19. **Copyright** footer

11.4 Sale Price, Concessions & Net SP

Row 5 shows three values for each comparable:

- **Sale Price** — the comparable's sale price from the data
- **Concessions** — the value from the column designated as `concessions` (0 if not designated)
- **Net SP** — an Excel formula: Sale Price − Concessions

The subject column shows "N/A" for sale price (since it is unknown) and any concessions value if available.

11.5 Grouped Rows (Location, Site, Age)

When certain special types are designated and those variables appear in the model, earthUI creates grouped rows that combine related variables:

Loc: Long | Lat | Area — Appears when any of `longitude`, `latitude`, or `area` are in the model. Shows factual values for each constituent variable, a combined Value Contribution (sum of the individual VCs), and for comps, an adjustment (subject combined VC − comp combined VC). Styled with light blue background on VC cells.

Site Size | Dimensions — Appears when `lot_size` or `site_dimensions` are in the model. Same structure as the Location group.

Actual Age | Effective Age — Appears when `actual_age` or `effective_age` are in the model. Same structure.

Variables consumed by grouped rows are **excluded from the individual model variable rows** below, preventing double-counting in the adjustment totals.

11.6 Model Variable Rows

Below the grouped rows (or directly below BASE VALUE if no grouped rows), one row per remaining model predictor shows:

- **Factual values** for subject and each comp (the actual data values)
- **Value Contribution (VC)** — the g-function's contribution to the predicted value for that row
- **Adjustment** (comps only) — subject VC minus comp VC

11.7 CQA|Residual & Residual Feature Rows

The **CQA|Residual** row shows each property's CQA score and contains a formula for the **remaining residual** — the portion of the residual not yet allocated to specific features. The formula is:

$$\text{Remaining Residual} = \text{Total Residual} - \sum(\text{Residual Feature VCs below})$$

Below this are **residual feature rows** — 5 named rows (Location, View/Appeal, Condition, Quality, Other) plus 6 blank rows. These are input cells where the appraiser enters value contributions for features not captured by the model. As the appraiser fills in values, the Remaining Residual formula automatically decreases.

For each comp, the adjustment column contains a formula: subject feature VC − comp feature VC.

11.8 Adjusted Sale Price Formula

The **Adjusted Sale Price** row contains Excel formulas:

- **Subject**: Sum of all Value Contribution cells from BASE VALUE through the last residual feature row. This represents the model's total prediction for the subject plus any appraiser-allocated residual features.
- **Comps**: Net SP + sum of all Adjustment cells from the first grouped or variable row through the last residual feature row. This gives the comparable's sale price after all regression-derived and appraiser-entered adjustments.

11.9 Sheet Protection & Editable Cells

Each sheet is **protected** to prevent accidental modification of formulas and data. The protection locks:

- All formula cells (Net SP, remaining residual, adjustments, adjusted sale price, etc.)
- All data cells (addresses, sale prices, factual values, value contributions from the model)
- Labels and headers

The only **unlocked** (editable) cells are the residual feature Value Contribution inputs — the cells under the CQA|Residual row where the appraiser enters breakdowns. These are styled with a light yellow background to indicate they are editable.

11.10 Working with the Grid in Excel

1. **Open the file** in Excel or a compatible spreadsheet application.
2. **Review the regression adjustments** — the model-derived adjustments are pre-populated and locked.
3. **Allocate the residual** — in the residual feature rows (yellow cells), enter your assessment of how much of the remaining residual is attributable to each feature (Location, View, Condition, Quality, etc.). The Remaining Residual formula will decrease as you allocate.
4. **Check the Adjusted Sale Price** — this formula automatically updates as you enter residual feature values.
5. **Multiple sheets** — if you selected more than 3 comps, navigate between sheets. Each sheet has the same subject in the left column with 3 different comps.

> ***Tip:*** *The goal is to allocate the CQA residual across the feature rows until the Remaining Residual is close to zero. This demonstrates that you can account for the difference between each comparable's regression estimate and its actual sale price.*

12 Downloading Reports

After fitting a model, report generation is a two-step Quarto workflow: **Generate Quarto Report** (sidebar section 9 in appraisal mode, 7 otherwise) writes a self-contained report bundle, and **Convert Quarto Report** (section 10) renders it — or any other Quarto file — to the final formats.

12.1 Generate Quarto Report

Clicking **Generate Quarto Report** writes a self-contained Quarto bundle under the project's output folder at `<base>_qmd/`, containing the populated `.qmd` source, all pre-generated plot assets (PNG and PDF), the report data, and a Word reference template. No rendering happens at this step — the bundle can be edited by hand, combined with other Quarto sources via Quarto includes, or previewed standalone with `quarto preview`.

12.2 Convert Quarto Report

Section 10 provides a file picker (defaulting to the current project's most recently generated bundle — but you can browse to any `.qmd` file) and format checkboxes:

- **HTML** — an `.html` file with KaTeX math rendering, embedded images, and a table of contents. Uses the Flatly Bootstrap theme and Roboto Condensed font.
- **Word** — a `.docx` file suitable for editing and distribution. Includes a table of contents, page numbers, and uses a custom reference template for consistent styling.
- **PDF** — typeset with LuaLaTeX for professional-quality output. Paper size follows the locale setting (Letter or A4). Includes landscape pages for large interaction matrices and uses Roboto Condensed with Latin Modern Math for equations. The PDF option is offered only when a LaTeX distribution is detected.

Click **Convert** to render the selected formats. Rendering runs in the background — a modal dialog shows elapsed time and Quarto progress while the app remains responsive. A notification confirms the file location on completion.

All operations (model fitting, data downloads, RCA calculations, sales grid generation, and report rendering) are logged with start/end timestamps and elapsed times to `<datafile>_earthui_log.txt` in the output folder, for troubleshooting and performance monitoring.

12.3 Report Contents

Every generated report includes the following sections:

1. **Dataset Description** — number of observations, target variable(s), predictors used, categorical predictors
2. **Model Specification** — degree, cross-validation status, number of terms, predictors in model
3. **Allowed Interactions** (degree ≥ 2 only) — the interaction matrix with checkmarks, formatted for the output type
4. **Results Summary** — R^2, GRSq, GCV, RSS, and CV R^2 (if cross-validation was enabled)
5. **Model Equation** — the complete Earth model equation in LaTeX notation
6. **Coefficients & Basis Functions** — table of all terms, their coefficients, and the basis functions that define them
7. **Variable Importance** — bar chart and ranked table of predictor importance scores
8. **g-Function Contributions** — plots for each g-function: line plots for univariate terms, 3D perspective and contour plots for bivariate interactions

9. **Correlation Matrix** — heatmap of predictor correlations
10. **Diagnostics** — Residuals vs Fitted, Normal Q-Q, and Actual vs Predicted plots
11. **ANOVA Decomposition** — table showing each basis function's contribution to RSS
12. **Earth Output** — raw text output from `earth::summary()`, including pruning pass details

For multi-response models, sections 4–8 and 10 are repeated for each target variable.

> ***Tip:*** *Reports require the* `quarto` *R package and a Quarto installation. For PDF output, a LaTeX distribution must also be installed (e.g., via* `tinytex::install_tinytex()`*) — otherwise the PDF checkbox is hidden.*

13 The Post Processing Modules: Elastic Net and GAM

The earthUI package ships two Post Processing Modules — an elastic net routine built on `glmnet` and a GAM routine built on `mgcv`. Each is treated in depth by its own article in this issue; this section explains what they are *for* in the earthUI workflow and how the three routines cooperate.

13.1 Why Support Fits?

The earth model produces the value conclusion. The Post Processing Modules exist to **corroborate** it: two additional estimates of the same subject, produced by methodologically different engines, fitted to the same project data. When an elastic net re-estimation of the earth basis and an independent GAM smooth arrive at materially the same value, the appraiser's conclusion no longer rests on a single model's assumptions — a meaningful advantage in review and litigation contexts. When they *disagree*, that disagreement is itself diagnostic, usually pointing at thin data, an over-flexible term, or an outlier comp worth revisiting. Neither module outranks the earth model; they are supporting evidence, deliberately presented without weighting.

13.2 Launching and Shared State

Each routine is a separate local Shiny application:

- `earthUI::launch_glmnet()` — the elastic net application, on port 7879.
- `earthUI::launch_mgcv()` — the GAM application, on port 7880.

All three applications read the same project tree and the same settings database, so the project you have open in earthUI is immediately available in the modules — along with the carry-forward fields earthUI saves for it: the Response (target), the Effective Date, the RCA subject-CQA type and value, and the living-area designation. You do not re-enter these; the modules read them so the same values flow through all three fits.

13.3 The Elastic Net Routine

The glmnet routine imports the earth model's basis functions — its hinges, interactions, and linear terms — as the columns of a design matrix and re-estimates them under lasso/elastic-net regularization (see "glmnetUI Import" above for the mechanics and the weight-column caveat). The result keeps earth's interpretable, geometrically meaningful terms while letting the regularizer shrink or drop the ones that do not earn their keep. See the glmnetUI article in this issue for the full treatment.

13.4 The GAM Routine

The mgcv routine goes the other direction: rather than reusing earth's basis, it fits fresh penalized smooth terms, using the earth model's knot locations as starting points (see "mgcvUI Auto-Export" above). Because the GAM re-derives the functional forms under a different smoothing paradigm, its agreement with the earth model is genuinely independent corroboration rather than an echo. Earth models with `degree` ≤ 2 are supported. See the mgcvUI article in this issue for the full treatment.

13.5 A Practical Workflow

1. Fit and refine the **earth model first**, in earthUI, until you are satisfied with it. Every successful fit (degree ≤ 2) auto-exports the `.rds` the modules consume.

2. Open the **glmnet** and **mgcv** routines against the same project, import the earth result, and fit. Work in one routine at a time — fits are long-running and the routines coordinate through the shared project, so the sensible rhythm is sequential, not parallel.
3. Iterate freely. Each application keeps its results while you switch between them, and nothing forces you to generate reports along the way.
4. Only when all three results satisfy you, return to each routine and generate its Quarto report. Each method registers its fit and value conclusion with the project's trilogy record, keeping the three reports traceable to one comparative run.

Tip: *The modules are optional at every level: earthUI's own workflow — fitting, RCA, Sales Grids, reports — is complete without them. Reach for them when a conclusion warrants corroboration: a difficult subject, a thin market segment, or an assignment likely to face review.*

14 Demo Dataset: Appraisal_1.csv

earthUI includes a demo MLS dataset for exploring the appraisal workflow. Load it programmatically with:

```
demo_file <- system.file("extdata", "Appraisal_1.csv", package = "earthUI")
df <- import_data(demo_file)
```

Or, in the Shiny app, copy it into a project's `in/` folder (or attach it as the initial data file when creating the project) and select it in Section 2 (Import Data).

14.1 Description

The file contains 1,501 residential sales plus 1 subject property in row 1 (1,502 rows in all) from a simulated MLS export. The data represents single-family home sales in a multi-area market with a range of property sizes, ages, and locations.

This is not real data, but is based on a realistic neighborhood in Northern California. All identification information has been altered or removed.

14.2 Columns

Column	Type	Special Type	Description
weight	numeric	—	Observation weight (0 = exclude from fitting)
id	numeric	display_only	Internal record ID
property_id	numeric	display_only	MLS property identifier
listing_id	character	display_only	MLS listing number
parcel_number	character	display_only	County assessor parcel number (APN)
street_address	character	display_only	Property address
city_name	character	display_only	City
postal_code	character	display_only	ZIP code
county_name	character	display_only	County
contract_date	Date	contract_date	Sale contract date (computes sale_age)
sale_age	numeric	—	Days from contract date to effective date (pre-computed)
coe_date	Date	display_only	Close of escrow date
listing_status	character	display_only	Listing status (e.g., "Sold")
sale_price	numeric	(target)	Sale price — response variable
rent	numeric	—	Monthly rent (for multi-target models)
list_price	numeric	display_only	Listing price
original_list_price	numeric	display_only	Original listing price
living_sqft	numeric	living_area	Gross living area in square feet
beds_total	integer	—	Number of bedrooms
baths_total	numeric	—	Total bath count (e.g., 2.5 = 2 full + 1 half)
lot_size	numeric	lot_size	Lot size in square feet
area_id	integer	area	MLS area identifier
area_text	character	display_only	Area name
age	numeric	actual_age	Property age in years
year_built	integer	display_only	Year of construction
latitude	numeric	latitude	Latitude (rounded to 3 dp for model)
longitude	numeric	longitude	Longitude (rounded to 3 dp for model)
latitude6	numeric	display_only	Full-precision latitude
longitude6	numeric	display_only	Full-precision longitude
garage_spaces	integer	—	Number of garage bays
fp_count	integer	—	Number of fireplaces
no_of_stories	numeric	—	Number of stories
style	character	—	Architectural style
view	character	—	View type (e.g., "Neighborhood", "Hills")
days_on_market	integer	dom	Days on market
listing_date	Date	listing_date	Listing date
sale_concessions	numeric	concessions	Seller concessions

14.3 Suggested Quick Start

1. Launch earthUI: `earthUI::launch()`
2. Create a project with Purpose **For Appraisal** in the New Project wizard, attaching `Appraisal_1.csv` as the initial data file
3. Select `Appraisal_1.csv` in Section 2 (Import Data)
4. Select `sale_price` as the target
5. Assign special types as shown in the table above
6. Include predictors: `sale_age`, `living_sqft`, `baths_total`, `lot_size`, `area_id` (as factor), `age`, `latitude`, `longitude`, `garage_spaces`
7. Set degree to 1, click **Fit Earth Model**
8. Download intermediate output (Step 6), review the CQA ranking
9. Compute RCA adjustments (Step 7) with a CQA score of ∼5.00
10. Generate the Sales Comparison Grid (Step 8)

Tip: *The demo dataset is a good way to explore all of earthUI's features before importing your own MLS data. The suggested predictors above typically produce a model with R^2 above 0.85.*

15 Experimental: Prolog Text Extraction & Derivation Rules

Version 0.10.0 introduces an **optional and genuinely experimental** facility for extracting model-ready features from the descriptive text columns of an MLS export — the "Public Remarks" column above all, but also agent remarks and comma-delimited feature lists such as room features or existing structures. The extraction engine is Prolog, and in particular **Definite Clause Grammars (DCGs)**: the rules that recognize phrases like "move in ready," "3 car garage," or "owned solar" in remark text are written as Prolog grammar rules.

This feature is still in development. It can be used *partially* for extraction today — the bundled grammar already recognizes a useful set of features — but adapting it to your own MLS's vocabulary requires you to read and write Prolog, and DCGs in particular. If that is not you, skip this chapter entirely: the feature is off by default, and the rest of earthUI works exactly the same without it.

Warning

This is an experimental feature under active development. Its extraction grammar is neither complete nor validated for production appraisal work — extracted columns must be reviewed against the source text before being relied upon in a model. Using the feature effectively requires working knowledge of Prolog and DCGs. Everything it does is additive and reversible; if anything misbehaves, turn the Settings toggle off and continue normally.

15.1 What It Does

MLS listings carry a great deal of value-relevant information in free text and in delimited list fields. This optional step turns that text into ordinary columns your regression can use, via two special column roles (Chapter 6):

- **`remarks` columns** are parsed by the Prolog DCG grammar into one feature column per extracted attribute. The column prefix is built from the first letters of the source column's name: `public_remarks` yields `pr_*` columns, `agent_remark` yields `ar_*`, and so on.
- **`list` columns** (e.g. `existing_structures = "Shed, Workshop"`) are split into one-hot `<col>_<value>` TRUE/FALSE columns. This part is pure R and needs no Prolog at all.

The bundled grammar (`inst/prolog/mls_remarks.pl`) currently extracts, among other things: a graded condition tier with a 1–5 `condition_score`; view flags and view types; kitchen and bath remodel flags; hardwood and other flooring; an upgrade list and count; garage spaces and garage area; pool, spa, and other outdoor features; fireplace count; outbuildings (shed, barn, workshop, casita, guest house, kennel, etc.) with sizes parsed from dimensions like "40x50"; RV parking and storage; solar ownership versus lease, battery storage; lot size (including acre-to-square-foot conversion); sale-context price amounts tied to a subject (price reduction, HOA dues, solar lease or payoff, Mello-Roos); location cues; and a highlights/concerns sentiment score in $[-1, 1]$.

15.2 Requirements: vProlog (Trealla), Not SWI-Prolog

Remarks parsing runs on the **vProlog** R package — a self-contained Trealla Prolog engine. It is not on CRAN; install it from the author's r-universe repository as shown in "Installing earthUI and vProlog" at the start of this article (prebuilt binaries for Windows and macOS, source at `github.com/wcraytor/vProlog`). No system Prolog installation (SWI-Prolog or otherwise) is required or used. If vProlog is absent, remarks parsing is skipped with a note while `list` splitting still runs. The helper `vProlog_available()` reports whether the engine and the bundled grammar are both present.

15.3 Enabling and Running It

1. Open **Settings** (gear) and tick **Enable Prolog / list column processing**. The toggle is off by default, carries an "advanced feature — for users with Prolog expertise" note, is remembered per project, and is shown in the Project Information dialog.
2. In the predictor table, set the **Special** role of the relevant columns to `remarks` or `list` (multiple columns may carry each role).
3. Use the **Execute Prolog Processing** section that sits between Sections 4 and 5 of the sidebar. It offers two steps:

1. Expand lists & remarks — runs the DCG grammar over each `remarks` column and one-hot-splits each `list` column. Expansion is *incremental*: only columns designated since the last run are processed, and the button is disabled once everything designated has already been expanded. A progress bar tracks the parse.

2. Execute derivation rules — applies your own Prolog `derive/2` rules over the expanded columns (see below). Rules are syntax-checked and dry-run validated before they run.

15.4 Derivation Rules

For each data row, earthUI asserts the row's columns as `cell(Name, Value)` facts and collects every solution of your `derive(Col, Val)` rules — each distinct `Col` becomes a new column, and the *first* value found per column wins, so a fallback clause may follow. Helper predicates (`count_true/2`, `count_false/2`, `count_true_prefix/2`, `cell_or/3`, `is_na/1`) support aggregation and NA handling, and `drop/1` facts discard columns after the rules run. For example:

```
% Merge a collinear cluster of solar flags into one boolean.
derive(solar_any, B) :-
    ( cell(pr_has_solar, true)
    ; cell(pr_solar_owned, true)
    ; cell(pr_has_battery, true) ) -> B = true ; B = false.

% Count amenities instead of OR-ing them (numeric column).
derive(amenity_count, N) :- count_true_prefix(pr_has_, N).

% Threshold -> boolean, with a fallback clause.
derive(big_outbuilding, true)  :- cell(pr_outbuilding_sqft, S), S >= 1000.
derive(big_outbuilding, false).

% Discard a raw granular column once consolidated.
drop(pr_price_solar_lease).
```

Rules live in a per-project `<project>_rules.pl` file, seeded from a fully commented template, and are edited via **Edit derivation rules...** in a floating, draggable code editor that can also be docked into the sidebar. A **Check** button validates the rules (syntax scan, load check, and a dry run against row 1 of your data) without executing them.

15.5 Versioned Active Data File

After either step, the augmented data frame is written to the project's input folder as `<original_name>_<YYMMDD_HHMMSS>.xlsx` and becomes the project's **active** data file — reopening the project loads the augmented data, so expansions are never recomputed, and the same file is what glmnetUI and mgcvUI consume. The raw remarks source columns are automatically held out of the model; the generated columns are *not* auto-included as predictors. Text parsing can produce many columns, so flag the ones you do not (yet) want in the regression as **Latent**, or ×-disable them (Chapter 6).

15.6 Programmatic Interface & Bundled Files

The whole pipeline is exported for batch use: `execute_prolog_processing(df, remarks_cols, list_cols, rules_text)` orchestrates the passes, built on `remarks_to_columns_()`, `split_list_column_()`, `apply_user_rules_()`, and `validate_rules_()`. The bundled Prolog sources under `inst/prolog/` are `mls_remarks.pl` (the DCG grammar and feature aggregation), `mls_analyze.pl` (a file-IO driver for batch parsing), `mls_rules.pl` (the helper predicates available to derivation rules), and `climate_regions_ca.pl` (below).

15.7 Related Experiment: Climate-Region Priors

A companion experimental facility addresses *thin-data* features — ones the regression cannot estimate because they appear in only a handful of comps (a swamp cooler, say). Three R helpers — `climate_region_for()`, `market_area_profile()`, and `climate_feature_priors()` — classify a project's location (county + city) into one of seven buyer-recognized California climate regions (a practical reduction of the California Energy Commission's 16 building-climate zones) and return ordinal contributory-value priors for HVAC and cooling features (heating, A/C, swamp cooler, whole-house fan, filtration) by region. The Prolog module `climate_regions_ca.pl` mirrors the same classification for use inside derivation rules; both sides are generated from one canonical R source so they cannot drift. The priors are intended to *inform* feature valuation and to serve as a plausibility check on estimated coefficients — region assignments (particularly for counties spanning several regions) should be reviewed by the appraiser.

15.7.1 Beyond California: Jurisdiction Packs and Market-Area Modules

California is the first jurisdiction, not the boundary. earthUI is used internationally, and the climate-prior knowledge is being restructured as **jurisdiction packs**: self-contained, data-shaped fact tables (zone definitions, admin-unit assignments, feature priors) keyed to the project system's country and admin-level codes, so a pack can describe a U.S. state, a German Land, or a small country outright. A companion layer of **market-area modules** captures local vocabulary and refined priors for a named market area.

Both are designed to be authored *in the field* by Valuation Engineers and appraisers, in their own language — and, realistically, with LLM assistance: the world does not have many Prolog programmers, so field artifacts are deliberately tables rather than programs. The pack format is a documented schema with a complete worked example (California itself), expressly so that an assistant such as Claude can draft a valid pack from a plain-language description of a market; earthUI generates the underlying Prolog from the pack, and a blocking validator (schema, coverage, referential-integrity, and dry-run checks) must pass before a pack can be enabled. Every field-authored pack carries mandatory provenance — the author of record, date, and the authority cited — so the responsible professional, not the drafting tool, stands behind the knowledge, and reports can cite the exact pack and version used.

> ***Tip:*** *A practical pattern: expand the remarks, consolidate the granular* `pr_*` *flags into a handful of derived counts or booleans with* `derive/2` *rules,* `drop/1` *the raw flags you consolidated, and keep only the derived columns as candidate predictors. This keeps the predictor list short and the collinearity manageable.*

References

Friedman, Jerome H. (1991). "Multivariate Adaptive Regression Splines". In: *The Annals of Statistics* 19.1, pp. 1–141. DOI: 10.1214/aos/1176347963.

Milborrow, Stephen (Oct. 1, 2024). *earth: Multivariate Adaptive Regression Splines*. R package. URL: https://CRAN.R-project.org/package=earth.

— (n.d.[a]). *Notes on the earth Package*. URL: http://www.milbo.org/doc/earth-notes.pdf.

— (n.d.[b]). *Variance Models in earth*. URL: http://www.milbo.org/doc/earth-varmod.pdf.