Election Forensics Toolkit: An R Package

Kirill Kalinin

September 24, 2025

1 BasicElectionForensics()

The `BasicElectionForensics` function is a comprehensive R tool designed to perform statistical analysis on election data to detect potential irregularities or fraud patterns. This function implements multiple forensic methods commonly used in election integrity research and provides both statistical results and significance testing through bootstrap methods.

```
BasicElectionForensics(data, Candidates, Level="National",
                       TotalReg, TotalVotes, Methods, R=1000, cores=2)
```

1.1 Input

Parameter	Type	Description
`data`	data.frame	The input dataset containing election results
`Candidates`	vector	Variable names referring to vote counts for candidates/parties
`Level`	string	Variable name depicting the level of analysis (default: "National")
`TotalReg`	string	Variable name for the total number of eligible voters
`TotalVotes`	string	Variable name for the total number of ballots cast
`Methods`	vector	List of forensic methods to apply (see Methods section)
`R`	numeric	Number of bootstrap simulations (default: 1000)
`cores`	numeric	Number of cores for parallel computing (default: 2)

1.2 Output

Output	Type	Description
`table`	`data.frame`	Numerical results of the election forensic tests
`tex`	`xtable`, `data.frame`	LaTeX-formatted results table suitable for academic use
`html`	`datatables`, `htmlwidget`	Interactive HTML table with color-coded significance
`sigMatrix`	`matrix`, `array`	Binary significance matrix for further statistical analysis

1.3 Statistical Distribution Tests

- `_2BL` - Second-digit mean test (Benford's Law analysis)
- `LastC` - Last-digit mean test for uniform distribution
- `P05s` - Percentage last-digit 0/5 indicator (for percentages)
- `C05s` - Count last-digit 0/5 indicator (for vote counts)

- `Skew` - Skewness measure (asymmetry from normal distribution)
- `Kurt` - Kurtosis measure (tail heaviness compared to normal distribution)
- `DipT` - Unimodality test (Hartigan's dip test)
- `Sobyanin` - Sobyanin-Sukhovolsky measure (turnout-vote share relationship)
- `Correlation` - Correlation coefficient between turnout and vote share

1.4 Key Features

- Uses nonparametric bootstrap with configurable number of simulations (R parameter)
- Provides confidence intervals for statistical significance
- Supports parallel processing for improved performance
- Works with various election data formats
- Handles missing data appropriately
- Supports multi-level analysis (e.g., national, regional, local)

1.5 Usage Example

```
library(EFToolkit)

# Load election data
dat <- read.csv(system.file("extdata/Albania2013.csv", package="EFToolkit"))

# Run forensics analysis
results <- BasicElectionForensics(
  dat,
  Candidates = c("C035", "C050"),
  Level = "Prefectures",
  TotalReg = "Registered",
  TotalVotes = "Ballots",
  Methods = c("P05s", "C05s", "_2BL", "Sobyanin",
              "DipT", "Skew", "Kurt", "Correlation"),
  cores = 2,
  R = 100  # Reduced for faster computation in example
)

# View results
print(results$table)
```

1.6 Interpretation Guidelines

See Table 3

2 BuildMap()

The `BuildMap` function is a specialized R visualization tool designed to create choropleth maps from election forensics analysis results. This function takes the output from `BasicElectionForensics()` and combines it with geographic data to produce spatial visualizations of statistical anomalies and patterns in election data.

```
BuildMap(eforensicsdata, geodata, Geoindex, Colorsig=FALSE, xlab="")
```

Search:

	Level	Candidate's Name	_2BL	P05s	C05s	Sobyanin	Skew	Kurt	DipT	Correlation	Obs
1	BERAT	Turnout	4.244	0.197	0.217	--	0.261	3.474	0.651	--	300
2			(3.92, 4.561)	(0.149, 0.254)	(0.162, 0.263)	--	(-0.121, 0.617)	(2.265, 4.374)	--	--	
3	DIBER	Turnout	4.167	0.221	0.225	--	1.685	4.954	0	--	258
4			(3.75, 4.52)	(0.173, 0.267)	(0.169, 0.273)	--	(1.314, 2.025)	(2.718, 6.392)	--	--	
5	DURRES	Turnout	4.572	0.173	0.194	--	1.354	3.211	0	--	444
6			(4.242, 4.812)	(0.131, 0.207)	(0.159, 0.231)	--	(1.078, 1.617)	(2.212, 3.956)	--	--	
7	ELBASAN	Turnout	4.345	0.217	0.196	--	1.5	4.086	0	--	516
8			(4.153, 4.632)	(0.184, 0.249)	(0.16, 0.226)	--	(1.245, 1.705)	(2.87, 4.84)	--	--	
9	FIER	Turnout	4.373	0.208	0.181	--	1.582	4.572	0	--	558
10			(4.105, 4.655)	(0.172, 0.239)	(0.153, 0.208)	--	(1.342, 1.777)	(3.435, 5.355)	--	--	
11	GJIROKASTER	Turnout	4.549	0.287	0.228	--	0.72	2.166	0	--	237
12			(4.008, 4.879)	(0.227, 0.342)	(0.175, 0.28)	--	(0.445, 0.954)	(1.529, 2.624)	--	--	

Figure 1: Basic Election Forensics Table

2.1 Input

Parameter	Type	Description
`eforensicsdata`	list	Output object from `BasicElectionForensics()` function
`geodata`	sf object	Spatial data frame containing geographic boundaries
`Geoindex`	string	Name of index variable from sf object used to merge geodata with forensics results
`Colorsig`	logical	If TRUE, only statistically significant estimates are mapped (default: FALSE)
`xlab`	string	X-axis label or subtitle for the maps

2.2 Output

Output	Type	Description
`figures`	`list`	Collection of generated plots (either `spplot` or `tmap` objects)
`Colorsig`	`logical`	Indicator of whether significance-based coloring was applied (`TRUE/FALSE`)
`shpdata`	`sf` (simple features)	Spatial dataset with election forensics results merged with geodata
`creationdate`	`POSIXct`	Timestamp of when the output was created

2.3 Key Features

2.3.1 Spatial Data Integration

- Seamlessly merges election forensics results with geographic boundaries
- Supports sf (Simple Features) spatial data format
- Handles missing data and geographic mismatches gracefully

2.3.2 Flexible Visualization Options

- **Standard Mode**: Displays all forensic values with continuous color scales

Table 3: Interpretation Guidelines

Test	No fraud	Interpretation
Second-digit mean (2BL)	4.187	Values close to 4.19 are consistent with Benford's Law. Systematic deviations (too low or too high) may indicate artificial rounding or human fabrication.
Last-digit mean (LastC)	4.5	Randomly distributed last digits should average 4.5. Substantial deviations suggest that numbers may not be uniformly random (e.g., preferences for certain digits).
Count of last-digit 0/5 ind. mean (C05s)	0.2	About 20% of values should end in 0 or 5. Excess frequency of 0s or 5s may reflect rounding or strategic reporting.
Perc. last-digit 0/5 ind. mean (P05s)	0.2	Same logic applies to percentages; deviations from 0.2 may signal manipulation of turnout or result percentages.
Skewness (Skew)	0	A symmetric distribution has skewness near zero. Positive skew indicates a longer right tail (many low but a few very high values); negative skew the opposite. Significant skewness may indicate anomalous clustering of results.
Kurtosis (Kurt)	3	A normal distribution has kurtosis of 3. Higher values (> 3) indicate peakedness (results too concentrated), while lower values (< 3) suggest excessive dispersion.
Unimodality test p-value (DipT)	> 0.05	A p-value greater than 0.05 supports unimodality (single peak). Values below 0.05 indicate multimodality, possibly reflecting a mixture of normal and manipulated results.
Sobyanin–Sukhovolsky	near 0	Captures the degree of association between turnout and vote share. Under normal conditions, this relationship should be weak or nonexistent; strong positive association may indicate manipulated results.
Correlation coefficient (Corr)	near 0	Measures the correlation between turnout and vote share across precincts. Values close to zero are expected in competitive elections; high positive correlations suggest manipulated results.

- **Significance Mode**: Highlights only statistically significant results
- Customizable color schemes using ColorBrewer palettes

2.3.3 Multi-Method Mapping

- Automatically generates maps for all forensic methods in the input data
- Creates separate visualizations for each candidate-method combination
- Supports multiple candidates and analysis levels simultaneously

2.4 Usage Example

```r
library(EFToolkit)
library(sf)

# Load election data
dat <- read.csv(system.file("extdata/Albania2013.csv", package="EFToolkit"))

# Run forensics analysis
eldata <- BasicElectionForensics(dat,
                                Candidates=c("C035", "C050"),
                                Level="Prefectures", TotalReg="Registered",
                                TotalVotes="Ballots",
                                Methods=c("P05s", "C05s"), R=100)

# Load geographic data
geodata <- st_read(system.file("extdata/Albania2013_prefectures.shp",
                        package="EFToolkit"), quiet = TRUE)

# Create maps
figures <- BuildMap(eforensicsdata=eldata, geodata=geodata, Geoindex="Level")

```

```
# Access individual maps
print(figures$figures$Turn_P05s)   # Turnout P05s method map
print(figures$figures$C035_P05s)   # Candidate C035 P05s method map
```

The function generates choropleth maps showing spatial patterns of election forensics indicators:

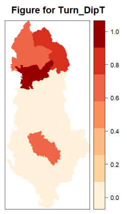


Image 1

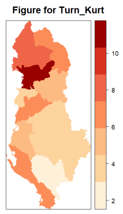


Image 2

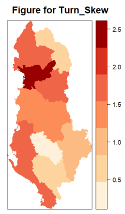


Image 3

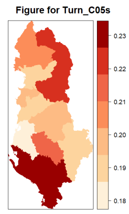


Image 4

2.5 Technical Implementation Details

2.5.1 Spatial Data Handling

- Uses `sf` package for modern spatial data processing
- Maintains coordinate reference systems throughout analysis
- Supports various geographic file formats (shapefile, GeoJSON, etc.)
- Implements ColorBrewer "OrRd" (Orange-Red) palette by default
- Provides good contrast for highlighting anomalies
- Color-blind friendly options available through ColorBrewer

2.5.2 Color Intensity Interpretation

- **Light Colors**: Values close to expected/normal ranges
- **Medium Colors**: Moderate deviations from expected patterns
- **Dark Colors**: Strong deviations suggesting potential irregularities
- **No Color/White**: Missing data or areas excluded from analysis

2.5.3 Spatial Pattern Analysis

- **Clustering**: Adjacent areas with similar colors may indicate systematic issues
- **Isolated Anomalies**: Single areas with extreme values warrant individual investigation
- **Border Effects**: Patterns along administrative boundaries may suggest institutional factors
- **Urban/Rural Differences**: Different patterns by area type are common

3 ClusterAnalysis()

The `ClusterAnalysis` function implements sophisticated spatial clustering tests to identify geographic patterns and anomalies in election forensics data. This function uses two complementary statistical methods - Getis-Ord Gi* and Local Moran's I - to detect spatial clusters of unusual values, helping researchers identify areas where election irregularities may be spatially correlated.

```
ClusterAnalysis(geodata, Vars, IndexCL=NULL, cores=2)
```

3.1 Input

Parameter	Type	Description
`geodata`	sf object/list	Spatial data within a list (supports both polygon and point sf objects) or BuildMap object
`Vars`	character vector	Variable names used for geographic clustering tests
`IndexCL`	string	Index variable name used for merging polygon and point sf objects (optional)
`cores`	numeric	Number of cores for parallel computing (default: 2)

3.2 Output

Output	Type	Description
`Moran's I for <Var>`	`ggplot`	A `ggplot` map visualizing local Moran's I cluster results for variable `<Var>`.
`Getis-Ord for <Var>`	`ggplot`	A `ggplot` map visualizing Getis-Ord G* hot- and cold-spots for variable `<Var>`.

3.3 Spatial Clustering Methods

3.3.1 Local Moran's I

- **Purpose**: Identifies local clusters and spatial outliers
- **Detects**: Four types of spatial associations:
  - **HH (High-High)**: High values surrounded by high values
  - **LL (Low-Low)**: Low values surrounded by low values
  - **HL (High-Low)**: High values surrounded by low values
  - **LH (Low-High)**: Low values surrounded by high values

3.3.2 Getis-Ord Gi* Statistic

- **Purpose**: Identifies hot spots and cold spots
- **Detects**: Two types of spatial concentrations:
  - **Hot Spots**: Areas with significantly high values clustered together
  - **Cold Spots**: Areas with significantly low values clustered together

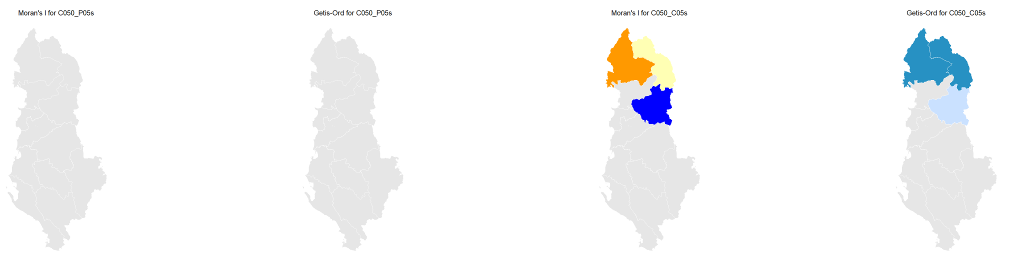
The function generates spatial cluster analysis maps showing different types of spatial patterns:

```r
library(EFToolkit)
library(sf)

# Load election data
dat <- read.csv(system.file("extdata/Albania2013.csv", package="EFToolkit"))

# Obtain election forensics estimates
eldata <- BasicElectionForensics(dat,
                                 Candidates=c("C035", "C050"),
                                 Level="Prefectures", TotalReg="Registered",
                                 TotalVotes="Ballots",
                                 Methods=c("P05s", "C05s"), R=100)

# Create the map with results
```



Moran P05s | Getis P05s | Moran C05s | Getis C05s

Figure 2: Spatial clustering analysis for Albanian election data - (1) Local Moran's I for P05s method, (2) Getis-Ord hot/cold spots for P05s, (3) Local Moran's I for C05s method, (4) Getis-Ord analysis for C05s method

```
geodata <- st_read(system.file("extdata/Albania2013_prefectures.shp",
                              package="EFToolkit"), quiet = TRUE)
figures <- BuildMap(eforensicsdata=eldata, geodata=geodata,
                   Geoindex="Level", Colorsig=FALSE)

# Using the mapped results, implement cluster analysis
cluster_results <- ClusterAnalysis(figures, Vars=c("C050_P05s", "C050_C05s"))

# View specific cluster maps
print(cluster_results[["Moran's␣I␣for␣C050_P05s"]])
print(cluster_results[["Getis-Ord␣for␣C050_P05s"]])
```

3.4 Key Features

3.4.1 Advanced Spatial Statistics

- **Permutation-based Testing**: Uses Monte Carlo permutation tests for statistical significance
- **False Discovery Rate Control**: Implements Benjamini-Hochberg FDR correction for multiple testing
- **Parallel Processing**: Supports multi-core computation for large datasets

3.4.2 Flexible Input Handling

- **Multiple Data Types**: Works with polygon and point spatial data
- **Integration Ready**: Seamlessly processes BuildMap function outputs
- **Missing Data Handling**: Robust treatment of missing or invalid data points

3.4.3 Comprehensive Visualization

- **Color-coded Results**: Uses standardized color schemes for easy interpretation
- **Multiple Significance Levels**: Shows results at 90%, 95%, and 99% confidence levels
- **Interactive Plotting**: Generates ggplot2 objects for further customization

3.5 Color Coding System

3.5.1 Local Moran's I Colors

Pattern	99% Level	95% Level	90% Level	Description
HH	Dark Red	Light Red	Pink	High-High clusters
LL	Dark Blue	Light Blue	Light Gray	Low-Low clusters

HL	Dark Orange	Light Orange	Yellow	High-Low outliers
LH	Dark Green	Light Green	Light Green	Low-High outliers
NS	Light Gray	Light Gray	Light Gray	Not significant

3.5.2 Getis-Ord Gi* Colors

Pattern	Description	Color
Hot Spot (99%)	Highly significant hot spot	Dark Red
Hot Spot (95%)	Significant hot spot	Medium Red
Hot Spot (90%)	Moderately significant hot spot	Light Red
Cold Spot (99%)	Highly significant cold spot	Dark Blue
Cold Spot (95%)	Significant cold spot	Medium Blue
Cold Spot (90%)	Moderately significant cold spot	Light Blue
Not Significant	No significant clustering	Light Gray

3.6 Interpretation Guidelines

3.6.1 Local Moran's I Results

- **High-High Clusters (Red)**: Areas with high forensic values surrounded by similar high values
  - *Interpretation*: Potential systematic irregularities in neighboring areas
  - *Action*: Priority areas for detailed investigation
- **Low-Low Clusters (Blue)**: Areas with low forensic values surrounded by similar low values
  - *Interpretation*: Regions with consistently normal patterns
  - *Action*: Lower priority for investigation
- **High-Low Outliers (Orange)**: High values surrounded by low values
  - *Interpretation*: Isolated anomalies or data quality issues
  - *Action*: Investigate for administrative or data collection problems
- **Low-High Outliers (Green)**: Low values surrounded by high values
  - *Interpretation*: Unusually clean areas within problematic regions
  - *Action*: Verify data accuracy or investigate protective factors

3.6.2 Getis-Ord Results

- **Hot Spots (Red Shades)**: Statistically significant concentrations of high values
  - *Interpretation*: Geographic clustering of potential irregularities
  - *Action*: Focus investigative resources on these areas
- **Cold Spots (Blue Shades)**: Statistically significant concentrations of low values
  - *Interpretation*: Areas with consistently normal electoral patterns
  - *Action*: Use as baseline comparisons or control areas

4 NonparamElectionForensics()

The `NonparamElectionForensics` function implements a revised version of Shpilkin's method for detecting election fraud through nonparametric analysis of vote distributions. This method identifies anomalous voting patterns by analyzing the relationship between turnout and candidate support, detecting artificial vote inflation through statistical modeling of "clean" electoral behavior.

```
NonparamElectionForensics(data, Candidates, CandidatesText=NULL,
                          MainCandidate, TotalReg, TotalVotes=NULL,
                          Level=NULL, MaxThreshold=0.8,
                          FigureName, setcolors=NULL,
                          precinctLevel=TRUE, computeSD=NULL,
                          sims=10, mode_search=list(npeaks=5, sortstr=TRUE,
                                                    minpeakdistance=1, pick_by="
                                                       height"),
                          man_turnout=NULL, grid_type="1D")
```

4.1 Input

Parameter	Type	Description
`data`	data.frame	Electoral data containing vote counts and registration information
`Candidates`	vector	Variable names for all candidates/parties in the election
`CandidatesText`	vector	Display names for candidates/parties (uses `Candidates` if NULL) (default: NULL)
`MainCandidate`	string	Variable name for main/incumbent candidate
`TotalReg`	string	Variable name for total number of eligible voters
`TotalVotes`	string	Variable name for total ballots cast (computed from candidates if NULL) (default: NULL)
`Level`	string	Variable for analysis level (default: "National")
`MaxThreshold`	numeric	Anomalous turnout threshold (default: 0.8)
`FigureName`	string	Title for generated figures
`setcolors`	vector	Custom color palette (random if NULL) (default: NULL)
`precinctLevel`	logical	Whether to compute precinct-level estimates (default: TRUE)
`computeSD`	string	Standard error method: "parametric" or "nonparametric" (default: NULL)
`sims`	numeric	Number of simulations for uncertainty estimation (default: 10)
`mode_search`	list	Clean peak search parameters
`man_turnout`	numeric	Manual clean peak turnout override (default: NULL)
`grid_type`	string	Estimation approach: "1D" or "2D" (default: "1D")

The `mode_search` parameter accepts a list with the following components:

Parameter	Description
`npeaks`	Maximum number of peaks to identify
`sortstr`	Whether to sort peaks by strength
`minpeakdistance`	Minimum distance between peaks

pick_by	Peak selection method: "area", "height", "cluster", "quantile", or "elipse"

4.2 Output

Output	Type	Description
list_graphs	list	Collection of generated plots (ggplot2/plotly objects)
base_stats	list	Basic fraud statistics for the whole dataset
sim_all_stats	list	Simulation statistics for the whole dataset (if computeSD specified)
sim_hetero_stats_base	data.frame	Base statistics for regional analyses (if Level != "National")
sim_hetero_stats_sims	data.frame	Simulation statistics for regional analyses (if Level != "National")
fraud_precinct_data	data.frame	Precinct-level fraud estimates with uncertainty measures
data	data.frame	Original input data with computed variables
Level	character	Analysis level used in the function
creationdate	POSIXct	Timestamp of when the output was created

4.2.1 Precinct-Level Fraud Data Structure

When `precinctLevel=TRUE`, the `fraud_precinct_data` component contains:

Column	Description
id	Unique precinct identifier
base.fraud.votes	Point estimate of fraudulent votes
sim.precinct_mean	Mean fraud estimate from simulations
sim.precinct_sd	Standard deviation of fraud estimates
sim.sig_all	Statistical significance indicator
precinct_mean_hetero	Regional-level fraud estimates (when Level != "National")

4.2.2 Primary Outputs

- **Official Turnout**: Reported voter participation rate
- **Real Turnout**: Estimated legitimate turnout (clean peak)
- **Official Support**: Reported candidate vote share
- **Real Support**: Estimated legitimate support in clean regions
- **Ballot Stuffing**: Votes added through turnout inflation
- **Ballot Switching**: Votes transferred between candidates
- **Total Fraud**: Combined fraudulent votes
- **Proportional Fraud**: Fraud as percentage of total votes

4.2.3 Uncertainty Quantification

When `computeSD` is specified, the function provides:

- **Parametric**: Assumes binomial distributions for vote generation
- **Nonparametric**: Uses bootstrap resampling for robust estimates

4.2.4 Clean Peak Detection Methods

- **”height”**: Selects clean peak with highest vote count
- **”area”**: Chooses clean peak with largest area under curve
- **”cluster”**: Uses clustering to identify clean peak
- **”quantile”**: Employs mixture models on turnout distribution for clean peak detection
- **”elipse”**: Uses robust covariance estimation for clean peak detection

4.2.5 Grid Types

- **1D Grid**: Uses 1D grid of turnout distribution to identify clean peak
- **2D Method**: Uses 2D grid of joint distribution of turnout and incumbent’s vote share

4.3 Example 1: National-Level Analysis with 1D Grid

```
library(EFToolkit)

# Load Russian 2000 election data
dat <- read.csv("electionfraud2000.csv")

# National analysis using 1D estimation method
res1 <- NonparamElectionForensics(dat,
                                  Candidates = paste("P", 1:12, sep=""),
                                  CandidatesText = c("Stanislav Govorukhin", "Umar Dzhabrailov",
                                                     "Vladimir Zhirinovsky", "Gennady Zuganov",
                                                     "Ella Pamfilova", "Alexei Podberezkin",
                                                     "Vladimir Putin", "Yuri Skuratov",
                                                     "Konstantin Titov", "Aman Tuleev",
                                                     "Grigorii Yavlinsky", "Against All"),
                                  MainCandidate = "P7",
                                  TotalReg = "NVoters",
                                  TotalVotes = "NValid",
                                  Level = "National",
                                  MaxThreshold = 0.8,
                                  mode_search = list(npeaks = 5, sortstr = TRUE,
                                                     minpeakdistance = 1, pick_by = "height"),
                                  FigureName = "Russian Presidential Elections, 2000",
                                  setcolors = c("royalblue2", "springgreen1","blue",
                                                "red", "green","brown2",
                                                "darkgreen", "yellow", "lawngreen",
                                                "purple","chartreuse1", "orange"),
                                  precinctLevel = TRUE,
                                  computeSD = "nonparametric",
                                  sims = 10,
                                  grid_type = "1D")

# Summary of precinct-level fraud estimates
total_fraud <- sum(res1$fraud_precinct_data$base.fraud.votes, na.rm=TRUE)
```

```
# Result: 2,685,246 fraudulent votes detected

# Statistically significant fraud only
significant_fraud <- sum(res1$fraud_precinct_data$sim.precinct_mean[
  res1$fraud_precinct_data$sim.sig_all==TRUE], na.rm=TRUE)
# Result: 811,501 significant fraudulent votes

> res1$base_stats
$`Whole dataset`
official_turnout     real_turnout official_support     real_support   ballot_
    stuffing ballot_switching
     6.820000e+01     6.700000e+01     5.330000e+01     5.200000e+01
         1.258868e+06     1.426378e+06
      total_fraud       prop_fraud
     2.685246e+06     6.990511e-02

# Display the table of region-level measures
View(round(res1$sim_hetero_stats_base, 3))
```

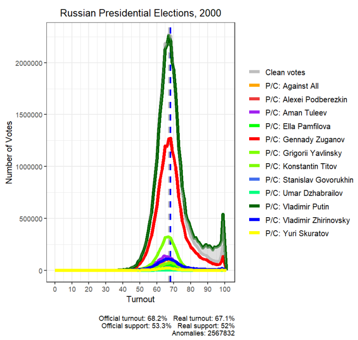


Image 1

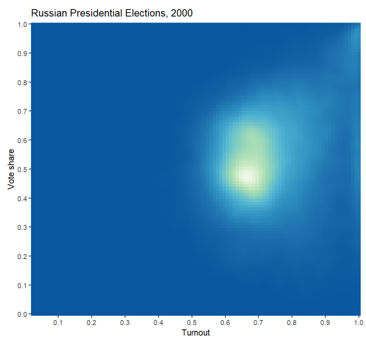


Image 2

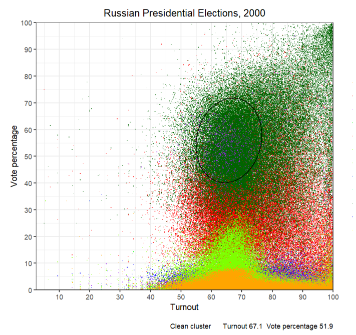


Image 3

	id	base.fraud.votes	sim.precinct_mean	sim.precinct_sd	sim.sig_all	base.sim.fraud.votes
27	Aginskiy Buryatskiy Avtonomnyy Okrug:10	144.0373	144.0373	51.48345	TRUE	152.9556
28	Aginskiy Buryatskiy Avtonomnyy Okrug:11	0	0	6.124084	FALSE	0
29	Aginskiy Buryatskiy Avtonomnyy Okrug:12	0	0	110.6984	FALSE	0
30	Aginskiy Buryatskiy Avtonomnyy Okrug:13	81.03211	81.03211	79.70118	FALSE	349.0933
31	Aginskiy Buryatskiy Avtonomnyy Okrug:14	61.11952	61.11952	67.72512	FALSE	0
32	Aginskiy Buryatskiy Avtonomnyy Okrug:15	92.00365	92.00365	46.36167	TRUE	158.9706
33	Aginskiy Buryatskiy Avtonomnyy Okrug:16	155.2359	155.2359	151.8857	FALSE	164.8476
34	Aginskiy Buryatskiy Avtonomnyy Okrug:17	94.40882	94.40882	133.2435	FALSE	190.4709
35	Aginskiy Buryatskiy Avtonomnyy Okrug:18	2.672858	2.672858	121.5221	FALSE	0
36	Aginskiy Buryatskiy Avtonomnyy Okrug:19	33.1881	33.1881	16.68935	TRUE	87.17307
37	Aginskiy Buryatskiy Avtonomnyy Okrug:2	0	0	82.82264	FALSE	0
38	Aginskiy Buryatskiy Avtonomnyy Okrug:20	0	0	93.76346	FALSE	0
39	Aginskiy Buryatskiy Avtonomnyy Okrug:21	0	0	72.17127	FALSE	0
40	Aginskiy Buryatskiy Avtonomnyy Okrug:22	44.09783	44.09783	25.82224	FALSE	0
41	Aginskiy Buryatskiy Avtonomnyy Okrug:23	0	0	95.87469	FALSE	12.75037
42	Aginskiy Buryatskiy Avtonomnyy Okrug:24	0	0	30.47656	FALSE	265.2888
43	Aginskiy Buryatskiy Avtonomnyy Okrug:25	66.37981	66.37981	15.20041	TRUE	187.8005
44	Aginskiy Buryatskiy Avtonomnyy Okrug:26	23.42156	23.42156	61.10772	FALSE	43.96371
45	Aginskiy Buryatskiy Avtonomnyy Okrug:27	18.32262	18.32262	184.3669	FALSE	27.25739
46	Aginskiy Buryatskiy Avtonomnyy Okrug:28	30.50753	30.50753	329.0753	FALSE	0
47	Aginskiy Buryatskiy Avtonomnyy Okrug:29	13.91468	13.91468	39.71865	FALSE	22.92176
48	Aginskiy Buryatskiy Avtonomnyy Okrug:3	0	0	86.5752	FALSE	0
49	Aginskiy Buryatskiy Avtonomnyy Okrug:30	71.03259	71.03259	24.98893	TRUE	133.3326
50	Aginskiy Buryatskiy Avtonomnyy Okrug:31	133.9505	133.9505	27.09151	TRUE	0
51	Aginskiy Buryatskiy Avtonomnyy Okrug:32	76.31489	76.31489	27.43527	TRUE	200.4515
52	Aginskiy Buryatskiy Avtonomnyy Okrug:33	43.79464	43.79464	93.04096	FALSE	72.14323
53	Aginskiy Buryatskiy Avtonomnyy Okrug:34	165.6622	165.6622	66.82878	TRUE	175.9195
54	Aginskiy Buryatskiy Avtonomnyy Okrug:35	0	0	133.141	FALSE	6.825516
55	Aginskiy Buryatskiy Avtonomnyy Okrug:36	0	0	111.9663	FALSE	0
56	Aginskiy Buryatskiy Avtonomnyy Okrug:37	0	0	35.80548	FALSE	0
57	Aginskiy Buryatskiy Avtonomnyy Okrug:38	87.85086	87.85086	19.92584	TRUE	230.7523
58	Aginskiy Buryatskiy Avtonomnyy Okrug:39	89.97062	89.97062	45.18136	TRUE	8.686122
59	Aginskiy Buryatskiy Avtonomnyy Okrug:4	0	0	71.50866	FALSE	0
60	Aginskiy Buryatskiy Avtonomnyy Okrug:40	20.29553	20.29553	83.30154	FALSE	0
61	Aginskiy Buryatskiy Avtonomnyy Okrug:41	69.32884	69.32884	103.2797	FALSE	103.1361
62	Aginskiy Buryatskiy Avtonomnyy Okrug:42	22.65131	22.65131	86.43285	FALSE	0
63	Aginskiy Buryatskiy Avtonomnyy Okrug:43	55.44521	55.44521	36.86485	FALSE	114.1747
64	Aginskiy Buryatskiy Avtonomnyy Okrug:44	83.0278	83.0278	57.56076	FALSE	170.9738

Figure 3: Precinct-level results

4.4 Example 2: Regional-Level Analysis with 1D Grid

```r
# Regional analysis across all federal subjects
res2 <- NonparamElectionForensics(dat,
                                  Candidates = paste("P", 1:12, sep=""),
                                  CandidatesText = c("Stanislav Govorukhin", "Umar Dzhabrailov",
                                                     "Vladimir Zhirinovsky", "Gennady Zuganov",
                                                     "Ella Pamfilova", "Alexei Podberezkin",
                                                     "Vladimir Putin", "Yuri Skuratov",
                                                     "Konstantin Titov", "Aman Tuleev",
                                                     "Grigorii Yavlinsky", "Against All"),
                                  MainCandidate = "P7",
                                  TotalReg = "NVoters",
                                  TotalVotes = "NValid",
                                  Level = "regname",  # Regional analysis
                                  MaxThreshold = 0.8,
                                  mode_search = list(npeaks = 5, sortstr = TRUE,
                                                     minpeakdistance = 1, pick_by = "height"),
                                  FigureName = "Russian Presidential Elections, 2000",
                                  setcolors = c("royalblue2", "springgreen1","blue",
                                                "red", "green","brown2",
                                                "darkgreen", "yellow", "lawngreen",
                                                "purple","chartreuse1", "orange"),
                                  precinctLevel = TRUE,
                                  computeSD = "nonparametric",
                                  sims = 10,
                                  grid_type = "1D")

# Regional fraud estimates
regional_fraud <- sum(res2$fraud_precinct_data$precinct_mean_hetero, na.rm=TRUE)
# Result: 1,693,742 fraudulent votes across regions

# Statistically significant regional fraud
significant_regional_fraud <- sum(res2$fraud_precinct_data$precinct_mean_hetero[
  res2$fraud_precinct_data$sim.sig_all==TRUE], na.rm=TRUE)
# Result: 1,125,151 significant fraudulent votes
```

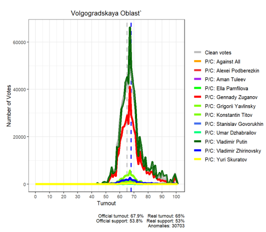


Image 1

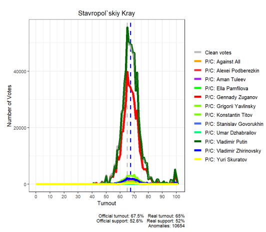


Image 2

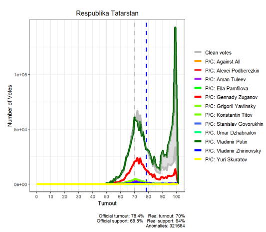


Image 3

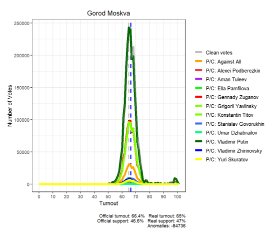


Image 4

4.5 Example 4: Analysis of Selected Regions with 2D Grid

	official_turnout	real_turnout	official_support	real_support	ballot_stuffing	ballot_switching	total_fraud	prop_fraud
`evenkiyskiy Avtonomnyy Okrug	55.2	44	62.7	57	796	-4.099723e-01	796	0.227819118
Aginskiy Buryatskiy Avtonomnyy Okrug	72.5	58	63.4	57	4773	3.284404e-01	4773	0.230813869
Altayskiy Kray	70.3	68	44.4	45	-16124	1.519160e-01	-16124	-0.030600528
Amurskaya Oblast`	67.6	63	49.9	51	-13681	-1.793228e-01	-13681	-0.058300130
Arkhangel`skaya Oblast`	68.6	66	60.3	59	14935	3.149228e-01	14935	0.034880027
Astrakhanskaya Oblast`	67.0	64	61.6	62	-2870	4.663774e-01	-2870	-0.009368736
Belgorodskaya Oblast`	72.7	70	48.1	47	18512	-2.778543e-01	18512	0.046170806
Bryanskaya Oblast`	69.8	68	43.4	44	-6119	1.172849e-01	-6119	-0.018503790
Chelyabinskaya Oblast`	67.5	66	49.5	50	-20379	-3.553078e-01	-20379	-0.022809053
Chitinskaya Oblast`	63.9	61	49.7	51	-11992	4.418638e-02	-11992	-0.046313675
Chukotskiy Avtonomnyy Okrug	73.5	71	67.7	60	1087	5.353024e+03	6440	0.292395005
shskaya Respublika - Chavash Respubliki	69.4	65	45.1	44	15301	-4.539740e-01	15301	0.050125140
Evreyskaya Avtonomnaya Oblast`	68.2	67	43.3	38	1674	4.806603e+03	6481	0.160357284
Gorod Moskva	66.4	65	46.6	47	-84736	-3.005550e-01	-84736	-0.039350797
Gorod Sankt-peterburg	66.0	58	62.7	64	-116952	4.710978e-01	-116952	-0.078660209
Irkutskaya Oblast`	64.5	61	50.6	50	19713	-4.754904e-01	19713	0.033653944
Ivanovskaya Oblast`	68.2	67	53.6	54	-978	2.262123e-01	-978	-0.002894338
Kabardino-balkarskaya Respublika	87.8	85	75.5	69	14306	9.263245e+04	106938	0.316031184
Kaliningradskaya Oblast`	66.3	62	61.2	63	-12990	-6.360557e-02	-12990	-0.072254978
Kaluzhskaya Oblast`	68.8	66	51.4	51	-761	4.770144e-01	-761	-0.002588435
Kamchatskaya Oblast`	63.3	62	49.1	45	3507	5.545969e+03	9053	0.111510747
Karachaevo-cherkesskaya Respublika	69.5	63	58.5	48	19855	3.185851e+04	51714	0.415667299
Kemerovskaya Oblast`	64.7	57	25.2	26	-18270	9.904484e-02	-18270	-0.052074426
Khabarovskiy Kray	64.5	62	50.0	51	-12856	-3.567119e-01	-12856	-0.035829447
Khanty-mansiyskiy Avtonomnyy Okrug	67.7	64	61.2	60	21364	-4.498280e-01	21364	0.057067300
Kirovskaya Oblast`	72.0	68	59.0	60	-9032	3.037944e-01	-9032	-0.017855872
Komi-permyatskiy Avtonomnyy Okrug	69.3	61	69.7	67	3897	3.867289e-01	3897	0.082614318

Figure 4: Region-level results

```
# Focus on specific regions of interest
selected_regions <- c("Respublika␣Dagestan", "Gorod␣Moskva",
                      "Samarskaya␣Oblast‘", "Volgogradskaya␣Oblast‘")
dat_subset <- dat[dat$regname %in% selected_regions,]

res4 <- NonparamElectionForensics(dat_subset,
                                 Candidates = paste("P", 1:12, sep=""),
                                 CandidatesText = c("Stanislav␣Govorukhin", "Umar␣Dzhabrailov",
                                                    "Vladimir␣Zhirinovsky", "Gennady␣Zuganov",
                                                    "Ella␣Pamfilova", "Alexei␣Podberezkin",
                                                    "Vladimir␣Putin", "Yuri␣Skuratov",
                                                    "Konstantin␣Titov", "Aman␣Tuleev",
                                                    "Grigorii␣Yavlinsky", "Against␣All"),
                                 MainCandidate = "P7",
                                 TotalReg = "NVoters",
                                 TotalVotes = "NValid",
                                 Level = "regname",
                                 MaxThreshold = 0.8,
                                 mode_search = list(npeaks = 5, sortstr = TRUE,
                                                    minpeakdistance = 1, pick_by = "height"),
                                 FigureName = "Russian␣Presidential␣Elections,␣2000",
                                 setcolors = c("royalblue2", "springgreen1","blue",
                                               "red", "green","brown2",
                                               "darkgreen", "yellow", "lawngreen",
                                               "purple","chartreuse1", "orange"),
                                 precinctLevel = TRUE,
                                 computeSD = "nonparametric",
                                 sims = 10,
                                 grid_type = "2D")
```

```
# Access regional comparison results
print(res4$stats_summary)
```

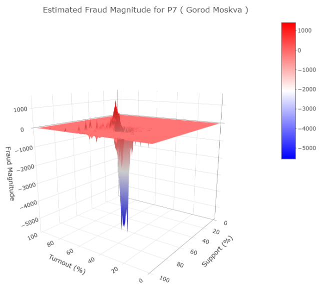


Image 1

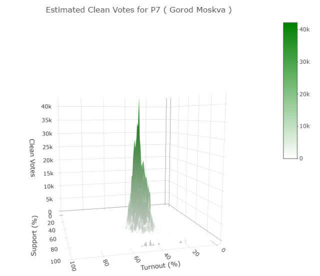


Image 2

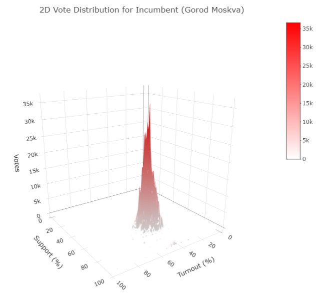


Image 3

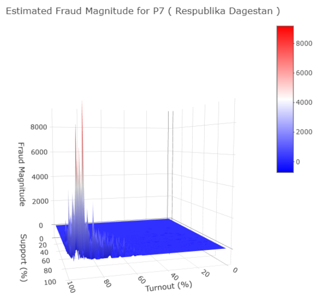


Image 4

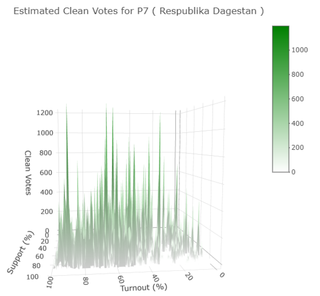


Image 5

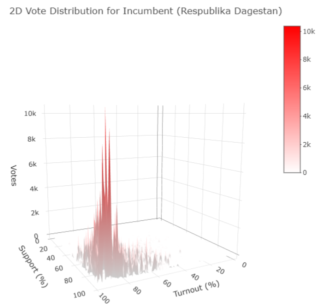


Image 6

4.6 Advanced Features

4.6.1 Multi-Level Analysis

When `Level` parameter specifies administrative units, the function:

- Performs analysis for the entire dataset
- Conducts separate analyses for each administrative unit
- Aggregates results across regions
- Provides comparative statistics

4.6.2 Robust Peak Detection

The algorithm implements multiple fallback strategies:

1. Primary method specified in `pick_by`
2. Alternative clustering approaches if primary fails
3. Simple peak detection as ultimate fallback
4. Manual override through `man_turnout` parameter

	id	base.fraud.votes	sim.precinct_mean	sim.precinct_sd	sim.sig_all	base.sim.fraud.votes	precinct_mean_hetero
1	`evenkiyskiy Avtonomnyy Okrug:1	155.8484	-1421.567	2054.079	FALSE	418.343	162.6206
2	`evenkiyskiy Avtonomnyy Okrug:10	25.30105	-427.5533	1561.618	FALSE	0	0
3	`evenkiyskiy Avtonomnyy Okrug:11	12.86312	135.1535	148.0531	FALSE	0	88.36314
4	`evenkiyskiy Avtonomnyy Okrug:12	4.562576	-791.3992	1913.307	FALSE	302.7445	164.3762
5	`evenkiyskiy Avtonomnyy Okrug:13	70.41162	-2101.368	2778.484	FALSE	0	98.72293
6	`evenkiyskiy Avtonomnyy Okrug:14	0	48.10323	194.0921	FALSE	52.77199	7.767163
7	`evenkiyskiy Avtonomnyy Okrug:15	0	94.90205	113.9037	FALSE	0	54.57691
8	`evenkiyskiy Avtonomnyy Okrug:16	103.1556	-889.7866	1914.451	FALSE	0	10.799
9	`evenkiyskiy Avtonomnyy Okrug:17	0	-282.1522	1158.442	FALSE	0	67.55037
10	`evenkiyskiy Avtonomnyy Okrug:18	0	-2581.953	3065.626	FALSE	113.8764	9.175749
11	`evenkiyskiy Avtonomnyy Okrug:19	28.21267	-958.3535	1648.795	FALSE	0	44.4601
12	`evenkiyskiy Avtonomnyy Okrug:2	141.1001	-196.8929	672.0185	FALSE	0	64.71237
13	`evenkiyskiy Avtonomnyy Okrug:20	22.39716	-2680.86	3391.231	FALSE	0	0
14	`evenkiyskiy Avtonomnyy Okrug:21	15.07341	-156.1077	261.9275	FALSE	0	0
15	`evenkiyskiy Avtonomnyy Okrug:22	22.81686	-1923.597	3938.503	FALSE	127.7638	17.53405
16	`evenkiyskiy Avtonomnyy Okrug:23	8.427413	-1156.796	1835.091	FALSE	0	17.30181
17	`evenkiyskiy Avtonomnyy Okrug:24	43.94621	-781.1408	1268.463	FALSE	0	51.7198
18	`evenkiyskiy Avtonomnyy Okrug:25	0	-4734.054	7018.571	FALSE	61.1044	24.4782
19	`evenkiyskiy Avtonomnyy Okrug:3	36.51768	7.870655	11.31684	FALSE	0	11.34118
20	`evenkiyskiy Avtonomnyy Okrug:4	18.39871	-1035.765	4512.612	FALSE	158.316	127.0222
21	`evenkiyskiy Avtonomnyy Okrug:5	0	82.71764	94.44965	FALSE	0	54.82581
22	`evenkiyskiy Avtonomnyy Okrug:6	0	61.76358	389.3413	FALSE	0	8.325508
23	`evenkiyskiy Avtonomnyy Okrug:7	0	-814.8789	1923.166	FALSE	0	1.753405
24	`evenkiyskiy Avtonomnyy Okrug:8	27.82745	81.65429	96.52667	FALSE	0	267.5425
25	`evenkiyskiy Avtonomnyy Okrug:9	0	132.5156	164.3393	FALSE	0	1.609483
26	Aginskiy Buryatskiy Avtonomnyy Okrug:1	133.0646	57.11153	69.36039	FALSE	0	94.85737
27	Aginskiy Buryatskiy Avtonomnyy Okrug:10	129.3876	-17.49128	102.7433	FALSE	852.2135	0
28	Aginskiy Buryatskiy Avtonomnyy Okrug:11	0	43.84897	62.63026	FALSE	46.0656	109.7196
29	Aginskiy Buryatskiy Avtonomnyy Okrug:12	0	63.37509	77.3835	FALSE	76.77599	406.2015
30	Aginskiy Buryatskiy Avtonomnyy Okrug:13	212.9548	-49.82751	372.391	FALSE	0	0
31	Aginskiy Buryatskiy Avtonomnyy Okrug:14	153.6347	55.67117	72.82713	FALSE	0	31.31473
32	Aginskiy Buryatskiy Avtonomnyy Okrug:15	41.11678	87.12984	94.31138	FALSE	0	50.1514
33	Aginskiy Buryatskiy Avtonomnyy Okrug:16	179.1718	-292.5687	1065.91	FALSE	0	134.6
34	Aginskiy Buryatskiy Avtonomnyy Okrug:17	158.943	31.27606	35.02158	FALSE	836.8583	8.610688
35	Aginskiy Buryatskiy Avtonomnyy Okrug:18	0	-478.2299	1341.243	FALSE	0	401.9558
36	Aginskiy Buryatskiy Avtonomnyy Okrug:19	54.52377	24.44194	32.80418	FALSE	0	50.6943
37	Aginskiy Buryatskiy Avtonomnyy Okrug:2	210.8207	20.86761	35.02686	FALSE	0	310.3425
38	Aginskiy Buryatskiy Avtonomnyy Okrug:20	0	-2.016188	10.21754	FALSE	0	0

Figure 5: Precinct-level results for selected regions analysis

4.6.3 Precinct-Level Estimation

When `precinctLevel=TRUE`, the function:

- Estimates fraud at individual precinct level
- Uses post-stratification for statistical adjustment
- Provides significance testing for precinct estimates
- Enables spatial analysis integration

4.7 Best Practices

4.7.1 Method Selection

1. **Start with 1D analysis** for initial exploration
2. **Use regional-level analysis** for heterogeneous countries
3. **Apply nonparametric uncertainty** for robust estimates
4. **Test multiple `pick_by` methods** for sensitivity analysis

4.7.2 Parameter Tuning

1. **Increase `sims`** for more precise uncertainty estimates
2. **Adjust** `MaxThreshold` based on country-specific context
3. **Experiment with** `mode_search` parameters for optimal peak detection
4. **Use custom** `setcolors` for publication-quality visualizations

4.7.3 Result Validation

1. **Compare across estimation methods** (1D vs 2D)
2. **Examine statistical significance** alongside magnitude
3. **Cross-validate with other forensic indicators**
4. **Consider substantive electoral context**

5 Finite Mixture Model()

`ComputeFiniteMixtureModel` - A legacy implementation of Walter Mebane's Finite Mixture Model for electoral data analysis.

The model uses Bayesian estimation techniques with EM-algorithm-like iterations to estimate the posterior probabilities of each precinct belonging to each fraud category.

> **Important Note**: This function is **legacy code** that is no longer actively maintained or supported. It may have dependencies on outdated packages or contain unoptimized algorithms.

```
ComputeFiniteMixtureModel(dat, MainCandidate = "Votes", TotalReg = "NVoters",
                          TotalVotes = "NValid", cores = 2, itstartmax = 1)
```

5.1 Input

Parameter	Type	Description
`dat`	data.frame	Electoral dataset containing voting data
`MainCandidate`	character	Variable name for the major/incumbent candidate votes (default: "Votes")
`TotalReg`	character	Variable name for total number of eligible voters (default: "NVoters")
`TotalVotes`	character	Variable name for total number of ballots cast (default: "NValid")
`cores`	integer	Number of cores for parallel computing (default: 2)
`itstartmax`	integer	Maximum number of iterations for optimization (default: 1)

5.2 Output

Returns a list containing FMM estimates with the following structure:

```
list(
  FF_null = matrix,     # Null model results (estimates and standard deviations)
  FFlist_null = list,   # Full null model output including posterior
      probabilities
  FF = matrix,          # Main model results (estimates and standard deviations)
  FFlist = list         # Full main model output including posterior
      probabilities
)
```

The output matrices contain the following parameters:

Parameter	Description
`incremental`	Proportion of incremental fraud
`extreme`	Proportion of extreme fraud
`alpha`	Fraud intensity parameter
`turnout`	Turnout rate parameter
`winprop`	Winning proportion parameter
`sigma`	Standard deviation for vote proportions
`stdAtt`	Standard deviation for attendance
`theta`	Convergence test parameter
`loglik`	Log-likelihood value
`df`	Degrees of freedom

5.3 Usage Example

```
library(EFToolkit)

# Load sample data
dat <- read.csv(system.file("extdata/ruspres2020.csv", package = "EFToolkit"))
dat <- subset(dat, select = c("region", "NVoters", "NValid", "Votes"))
datc <- dat[dat$region == "Volgogradskaya␣Oblast‘", ]

# Run FMM analysis (commented out due to long computation time)
# res <- ComputeFiniteMixtureModel(datc,
#                                  MainCandidate = "Votes",
#                                  TotalReg = "NVoters",
#                                  TotalVotes = "NValid")
```

5.4 Key Features

- **Mixture Modeling**: Implements a finite mixture model with multiple fraud components
- **Parallel Computing**: Supports multi-core processing for faster computation
- **Robust Estimation**: Uses genetic algorithms (`rgenoud`) for parameter optimization
- **Statistical Inference**: Provides estimates with standard errors

5.5 Limitations & Considerations

1. **Computational Intensity**: The function can be very slow for large datasets
2. **Legacy Status**: No longer actively supported or maintained
3. **Algorithm Complexity**: Implements sophisticated statistical models that may require domain expertise to interpret
4. **Parameter Sensitivity**: Results may be sensitive to starting values and iteration limits

6 Klimek Model()

`ComputeKlimekModel` - Implements the Klimek et al. (2012) simulation-based method for detecting electoral anomalies through histogram analysis of vote distributions.

Legacy Function Notice: This function is **legacy code** that is no longer actively maintained or supported. It remains available for historical reference and research reproducibility but may have dependencies on outdated packages or contain unoptimized algorithms.

6.1 Key Features:

- **Histogram-based analysis**: Compares observed vote distributions with simulated ones
- **Fraud simulation**: Models incremental and extreme fraud scenarios
- **Parameter estimation**: Estimates fraud parameters through iterative simulation
- **Statistical testing**: Uses chi-square goodness-of-fit tests to evaluate model fit

```
ComputeKlimekModel(data, Candidates, Level = "National", TotalReg, TotalVotes,
                   R = 1000, cores = 2)
```

6.2 Input

Parameter	Description
data	Data frame containing electoral data
Candidates	Variable name(s) for vote counts of candidates/parties
Level	Variable indicating geographic level of analysis (default: "National")
TotalReg	Variable name for total number of eligible voters
TotalVotes	Variable name for total number of ballots cast
R	Number of simulations to run (default: 1000)
cores	Number of CPU cores for parallel computation (default: 2)

6.3 Output

Column	Description
Level	Geographic level of analysis
Candidate	Candidate/party name
KSimI	Estimated proportion of incremental fraud
KSimE	Estimated proportion of extreme fraud
KSimalpha	Fraud intensity parameter
KSimturnout	Estimated turnout rate parameter
KSimwinprop	Estimated winning proportion parameter
KSimsigma	Standard deviation for vote proportions
KSimstdAtt	Standard deviation for attendance rates
KSimtheta	Convergence test parameter
Obs	Number of observations used in analysis

6.4 Usage Example

```r
library(EFToolkit)

# Load sample data
dat <- read.csv(system.file("Albania2013.csv", package = "EFToolkit"))

# Run Klimek analysis with reduced simulations for speed
klimek <- ComputeKlimekModel(dat,
                              Candidates = "C050",
                              Level = "National",
                              TotalReg = "Registered",
                              TotalVotes = "Ballots",
                              cores = 1,
                              R = 100)

# Access results
print(klimek$table)      # Data frame with results
print(klimek$html)       # HTML formatted table
print(klimek$tex)        # LaTeX formatted table
```

The function computes several goodness-of-fit measures:

- **Winner.HFit.Klimek**: Histogram fit statistic for winning candidate
- **Winner.HFit.chi2**: Chi-square histogram fit
- **Winner.Fit.chi2**: Chi-square vote count fit
- **Overall chi2**: Comprehensive model fit statistic

6.5 Key Functions:

- `Estimate()`: Initial parameter estimation from vote distributions
- `Sim_Vote()`: Simulates electoral outcomes under fraud scenarios
- `Sim.Histo()`: Generates histogram distributions from simulated data
- `Iteration_sim()`: Main optimization loop for parameter estimation

6.6 Simulation Parameters:

- **f1range**: Incremental fraud proportion range (0.0 to 1.0)
- **f2range**: Extreme fraud proportion range (0.0 to 0.3)
- **arange**: Fraud intensity range (0.5 to 1.0)
- **iterations**: Number of optimization iterations (default: 10)

6.7 Interpretation Guidelines

- **Low fraud estimates** (KSimI, KSimE near 0): Suggest clean election
- **High incremental fraud**: Indicates widespread small-scale manipulation
- **High extreme fraud**: Suggests concentrated large-scale fraud
- **Model fit statistics**: Lower values indicate better model fit to observed data

7 Installation

7.1 Prerequisites

- R version 4.0.0 or higher
- Required R packages: `dplyr`, `ggplot2`, `sf`, `spdep`, `spatialreg`, `sp`, `pracma`, `plotly`, `pbapply`, `DT`, `xtable`, `parallel`, `diptest`, `poweRlaw`, `RColorBrewer`, `maptools`, `spatstat`, `sparr`, `sparr`, `spatstat`, `spatstat.core`, `spatstat.linnet`, `spatstat.geom`, `spatstat.data`, `spatstat.utils`, `spatstat.sparse`, `spatstat`, `spatstat.core`, `spatstat.linnet`, `spatstat.geom`, `spatstat.data`, `spatstat.utils`, `spatstat.sparse`

7.2 Installation from GitHub

```r
# Install devtools if not already installed
if (!require(devtools)) install.packages("devtools")

# Install the package
devtools::install_github("kalinin1/EFToolkit")
```

7.3 Installation from Source

```r
# Download the package source and install
install.packages("path/to/EFToolkit_1.0.tar.gz", repos=NULL, type="source")
```

8 License

This package is distributed under the MIT License.

9 Citation

When using this package in research, please cite:

Kalinin, K., & Mebane, W. (2025). Election Forensics Toolkit: Statistical Methods for Detecting Election Irregularities. R package version 1.0.

10 Contact

- Kirill Kalinin: kalinin@umich.edu
- Walter Mebane: mebane@umich.edu