Contents

OH Stats: Complete Learning & Reference Guide

A comprehensive guide to statistical analysis of Occupational Health data

For beginners learning statistics AND practitioners needing a reference

---

How to Use This Guide

Navigation Guide

Learning statistics? → Start with Part I, read sequentially

Running an analysis? → Jump to Part II for step-by-step workflow

Troubleshooting? → Part IV has solutions to common problems

Writing a paper? → Part III has reporting templates

PART I: Statistical Foundations

Everything you need to understand WHY we use these methods

---

1. What Are We Analyzing?

The OH Profile: Multi-Modal Health Data

An OH (Occupational Health) profile contains multiple types of data collected from workers:

Data Type	Examples	Measurement Scale
Sensor metrics	EMG, accelerometer, heart rate, noise	Continuous (numeric)
Questionnaires	COPSOQ, MUEQ, ROSA, IPAQ, OSPAQ	Ordinal or Continuous
Daily self-reports	Workload, pain ratings (NPRS)	Ordinal (Likert 1-5, NPRS 0-10)
Environmental	Temperature, humidity	Continuous

The Unit of Analysis: Subject x Day

Key Concept

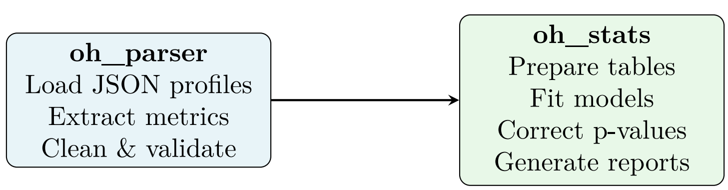
Your primary unit of analysis is **one subject on one day**.

Each row in your analysis = one person's measurements for one day

Key points:

- Each subject contributes **daily aggregated metrics** (e.g., average EMG over the whole day)
- The data are **naturally unbalanced** – some subjects have 3 days, others have 5
- You analyze **each modality separately** (EMG models don't mix with posture or HR models)

The Two-Package Ecosystem



---

2. Why T-Tests Don't Work Here

The Setup

Imagine you're studying muscle fatigue in office workers. You measure their EMG (muscle activity) every day for a week. Your question: **Does muscle activity change over the week?**

Your data looks like this:

```
Subject  Day  EMG_value
-------  ---  ---------
Alice     1       10.2
Alice     2        9.8
Alice     3        8.5
Bob       1       15.1
Bob       2       14.8
...
```

The Independence Problem

The Problem

Alice's measurements are all related to each other. If Alice naturally has low muscle activity, ALL her measurements will be lower.

T-tests assume **every measurement is independent** – like flipping a coin. But Alice's Day 2 is NOT independent from Alice's Day 1.

Mathematically: The independence assumption fails because $\mathrm{Cov}(Y_{\mathrm{Alice,Day1}}, Y_{\mathrm{Alice,Day2}}) \neq 0$.

What Happens If We Ignore This?

Problem	Effect
Inflated sample size	We count 320 observations, but really have ~37 independent subjects
Too-small p-values	Standard errors underestimated, leading to false confidence
False discoveries	We "find" effects that aren't real
Unreproducible results	Different samples give wildly different answers

The Coin Flip Analogy

Analogy

Imagine you flip a coin 10 times, but you **count each flip 10 times**. You now have "100 observations" but really only 10 independent flips. If you got 6 heads in those 10 real flips, you'd report 60 heads in "100 flips" – and wrongly conclude the coin is biased!

That's exactly what happens when you use t-tests on repeated measures data.

---

3. Linear Mixed Models: The Solution

The Key Idea

Linear Mixed Models (LMMs) solve this by recognizing **TWO sources of variation**:

$$\boxed{\textbf{Total variation} = \text{Between-subject variation} + \text{Within-subject variation}}$$

Component	What it represents
Between-subject	Alice vs Bob differences (personal baselines)
Within-subject	Day-to-day changes for the same person

The Intuitive Model

```
EMG_value = Overall_average + Day_effect + Subject's_baseline + Noise
```

Component	What it represents
Overall_average	The typical EMG value across everyone
Day_effect	How much Day 2, 3, 4, etc. differ from Day 1 ← *this is what we test!*
Subject's_baseline	Alice is naturally 3 units lower, Bob is 5 units higher, etc.
Random_noise	Unexplained day-to-day fluctuations

The Statistical Formula

For a continuous outcome Y for subject i on day j:

$$Y_{ij} = \beta_0 + \beta_{\text{day}(j)} + u_i + \varepsilon_{ij}$$

Where:

- β_0 = grand mean (intercept)
- $\beta_{\text{day}(j)}$ = effect of day (Day2 vs Day1, Day3 vs Day1, …)
- $u_i \sim N(0, \sigma_u^2)$ = subject-specific random intercept
- $\varepsilon_{ij} \sim N(0, \sigma^2)$ = residual error

The ICC: Measuring Clustering

The **Intraclass Correlation (ICC)** tells you what proportion of total variation is due to between-subject differences:

$$\text{ICC} = \frac{\sigma_u^2}{\sigma_u^2 + \sigma^2} = \frac{\text{Between-subject variance}}{\text{Total variance}}$$

How to interpret ICC:

ICC Value	Meaning	Implication
0.0 – 0.2	Low clustering	Subjects are similar; most variation is day-to-day
0.2 – 0.5	Moderate	Both sources of variation matter
0.5 – 0.8	Strong	Who you are matters a lot
0.8 – 1.0	Very strong	Almost all variation is between people

Important

In our EMG data, ICC is typically **0.4–0.6**. If ICC is high, you REALLY need mixed models. Using t-tests would give wrong answers.

---

4. Understanding P-Values and Significance

What a P-Value Actually Means

The **p-value** answers: *"If there were NO real effect, how often would we see data this extreme?"*

p-value	Interpretation
$p < 0.01$	Strong evidence of a real effect
$p < 0.05$	Moderate evidence (conventional threshold)
$p < 0.10$	Weak evidence, worth noting but not conclusive
$p > 0.10$	Not enough evidence to claim an effect

Common Misinterpretations

Common Mistakes

$p < 0.05$ does **NOT** mean "95% sure the effect is real."

What people think	Reality
"95% chance the effect is real"	**WRONG**
"5% chance this is a false positive"	**WRONG**

What people think	Reality
"If there's no effect, we'd see this data only 5% of the time"	**CORRECT**

Statistical vs. Practical Significance

A **statistically significant** result ($p < 0.05$) tells you the effect is probably real. It does NOT tell you the effect is **large enough to matter**.

Example

"Day 4 is 0.2 %MVC lower than Day 1" might be significant ($p = 0.04$) with large samples. But is 0.2 %MVC clinically meaningful? That's a separate question!

Always report effect sizes alongside p-values.

---

5. Effect Sizes: How Big Is the Effect?

Raw Units vs. Standardized

Approach	Example	When to use
Raw units	"Day 4 is 1.93 %MVC lower"	Primary reporting; clinically interpretable
Cohen's d	"d = 0.28 (small effect)"	Cross-study comparison; standardized

Cohen's d for Mixed Models

In LMMs, variance is decomposed into components, so there's no single "pooled SD." We use **residual-standardized effect size**:

$$d = \frac{\Delta}{\sigma_{\text{residual}}}$$

Where Δ = contrast estimate and σ_{residual} = square root of residual variance.

Cohen's d Interpretation

d	
< 0.2	Negligible
0.2 – 0.5	Small
0.5 – 0.8	Medium

d	
≥ 0.8	Large

Recommendation

Always report raw-unit effects with confidence intervals as the primary result. Cohen's d is supplementary for readers who want standardized comparisons across studies.

---

6. The Multiple Testing Problem

The Problem

You're analyzing 20 different EMG outcomes. Even if NONE of them have real effects, you'll probably find at least one "significant" result just by chance!

Why? If $p < 0.05$ means "5% chance when there's no effect," then:

- Test 1 outcome: 5% chance of false positive
- Test 20 outcomes: $1 - (0.95)^{20} \approx$ **64% chance** of AT LEAST ONE false positive!

The Solution: Correction Methods

Method	Controls	When to use
FDR (False Discovery Rate)	Expected proportion of false discoveries	Exploratory analysis, many outcomes
FWER (Family-Wise Error Rate)	Chance of ANY false positive	Confirmatory, few primary outcomes

The Two-Layer Strategy

Our pipeline uses a **two-layer correction**:

Two-Layer Correction Strategy

Layer 1 (across outcomes): FDR on LRT p-values
"Which outcomes show any day effect?"

✔ EMG_intensity.mean_percent_mvc: p_adj = 0.005
✔ EMG_apdf.active.p50: p_adj = 0.04
✘ EMG_apdf.rest.p10: p_adj = 0.12

Layer 2 (within outcome): Holm on post-hoc contrasts
"Which specific days differ?" (only for outcomes that passed Layer 1)

Critical: Which P-Value Feeds FDR?

Critical Distinction

When you see the coefficients table with individual p-values for Day 2, Day 3, etc., we do **NOT** use these for FDR correction.

Instead, we use the **omnibus Likelihood Ratio Test (LRT)** p-value, which asks: "Does including 'day' improve the model AT ALL?"

```
# Access the LRT p-value (this feeds FDR!)
print(result['fit_stats']['lrt_pvalue'])
```

Why the LRT, not coefficient p-values?

- Coefficient p-values test "Day 2 vs Day 1", "Day 3 vs Day 1", etc. – many tests per outcome!
- LRT asks ONE question per outcome: "Is there ANY day effect?"
- FDR needs ONE p-value per outcome to work correctly

---

7. Model Assumptions and Diagnostics

The Main Assumptions

Assumption	What it means	How to check
Residuals ~ Normal	"Leftovers" should be bell-shaped	QQ plot, Shapiro-Wilk
Constant variance	Spread doesn't change with fitted values	Residuals vs. Fitted plot
Independence within clusters	After accounting for subjects, variation is random	Study design

Don't Panic About Violations!

Good News

LMMs are fairly robust to mild assumption violations.

Situation	What to do
Shapiro-Wilk $p < 0.05$ but QQ plot looks OK	Probably fine, especially with $N > 30$
Moderate skewness ($\|skew\| < 1$)	Usually OK; consider transform if severe
A few outliers	Investigate them; run sensitivity analysis

Visual Diagnostics (Most Important!)

```python
import matplotlib.pyplot as plt
from scipy import stats

fig, axes = plt.subplots(1, 2, figsize=(10, 4))

# QQ Plot: Points should follow the diagonal line
stats.probplot(diag['standardized'], dist="norm", plot=axes[0])
axes[0].set_title("QQ Plot (should be a straight line)")

# Residuals vs Fitted: Should be a random cloud around zero
axes[1].scatter(diag['fitted'], diag['residuals'], alpha=0.5)
axes[1].axhline(y=0, color='r', linestyle='--')
axes[1].set_xlabel("Fitted Values")
axes[1].set_ylabel("Residuals")
axes[1].set_title("Residuals vs Fitted (should be random cloud)")

plt.tight_layout()
plt.show()
```

PART II: The Analysis Workflow

Step-by-step guide to running your analysis

---

8. Step 1: Load and Discover Your Data

Load Profiles

```
from oh_parser import load_profiles
from oh_stats import get_profile_summary, discover_sensors, discover_questionnaires

# Load the OH profiles
profiles = load_profiles("/path/to/OH_profiles")

# FIRST: See what data is available (recommended!)
print(get_profile_summary(profiles))
```

Example output:

```
OH Profile Summary (42 subjects)
==================================================

SENSOR DATA:
  emg: 15 metrics
  heart_rate: 8 metrics
  noise: 6 metrics

SINGLE-INSTANCE QUESTIONNAIRES:
  personal: 31 fields
  biomechanical: 73 fields

DAILY QUESTIONNAIRES:
  workload: 6 fields
  pain: 12 fields
```

Explore Specific Sensors

```
# What EMG metrics are available?
sensors = discover_sensors(profiles)
print(sensors['emg'])
# ['EMG_intensity', 'EMG_apdf', 'EMG_muscular_rest', ...]

# What questionnaires?
quests = discover_questionnaires(profiles)
print(quests['single_instance'].keys())
# ['copsoq', 'mueq', 'rosa', 'ipaq', 'ospaq']
```

---

9. Step 2: Prepare Your Data

The AnalysisDataset Container

AnalysisDataset Structure

All analysis functions expect an **AnalysisDataset** – a standardized container:

- `ds['data']` – The actual DataFrame (long format)
- `ds['outcome_vars']` – List of outcome column names
- `ds['id_var']` – Clustering variable (usually 'subject_id')
- `ds['time_var']` – Time variable (usually 'day_index')
- `ds['grouping_vars']` – Additional grouping (e.g., ["side"])

Two Ways to Prepare Data

Recommended Workflow

Option A: Convenience wrappers (simple cases)
Use `prepare_daily_emg(profiles)` when you want standard extraction.

Option B: From pre-extracted DataFrame (more control)
Use `prepare_from_dataframe(df)` when you've already extracted data with oh_parser and want to customize filtering/transformations.

From Pre-Extracted DataFrame (Maximum Control)

If you've already extracted data with oh_parser, use `prepare_from_dataframe()`:

```python
from oh_parser import extract_nested
from oh_stats import prepare_from_dataframe, fit_lmm

# Step 1: Extract with oh_parser (you control this)
df = extract_nested(
    profiles,
    base_path="sensor_metrics.emg",
    level_names=["date", "level", "side"],
    value_paths=["EMG_intensity.*"],
    flatten_values=True,
)

# Step 2: Apply your custom filtering
df = df[df["level"] == "EMG_daily_metrics"]
df = df.drop(columns=["level"])

# Step 3: Convert to AnalysisDataset (no redundant extraction!)
ds = prepare_from_dataframe(df, sensor="emg", side="average")

# Step 4: Use oh_stats as normal
result = fit_lmm(ds, "EMG_intensity.mean_percent_mvc")
```

Prepare EMG Data (Convenience Wrapper)

```python
from oh_stats import prepare_daily_emg

# Keep both sides as separate rows
ds = prepare_daily_emg(profiles, side="both")

# Or average left/right (simpler - RECOMMENDED)
ds = prepare_daily_emg(profiles, side="average")
```

Side handling options:

Strategy	Effect	When to Use
`"both"`	Left and right as separate rows	When laterality is of interest
`"average"`	Mean of left/right	When laterality is nuisance (**recommended**)
`"left"` / `"right"`	Keep only one side	When sides have different meaning

Prepare Any Sensor (Generic)

```python
from oh_stats import prepare_sensor_data

# Heart rate data
hr_ds = prepare_sensor_data(
    profiles,
    sensor="heart_rate",
    base_path="sensor_metrics.heart_rate",
    level_names=["date"],
    value_paths=["HR_BPM_stats.*", "HR_ratio_stats.*"],
)

# Noise data
noise_ds = prepare_sensor_data(
    profiles,
    sensor="noise",
    base_path="sensor_metrics.noise",
    level_names=["date"],
    value_paths=["Noise_statistics.*"],
)
```

Prepare Questionnaire Data

```python
from oh_stats import (
    prepare_baseline_questionnaires,
    prepare_daily_pain,
    prepare_daily_workload
)

# Single-instance baseline questionnaires
```

```
baseline_ds = prepare_baseline_questionnaires(profiles, questionnaire_type="copsoq")

# Daily repeated measures
pain_ds = prepare_daily_pain(profiles)
workload_ds = prepare_daily_workload(profiles)
```

Prepare Unified Daily Metrics (Sensors + Workload)

```
from oh_stats import prepare_daily_metrics

# Unified daily dataset with HR, noise, HAR, EMG, and workload
daily_ds = prepare_daily_metrics(profiles)
print(daily_ds["data"].head())
```

Included metrics (when available):

- Workload daily mean (5 fixed items)
- Human activities: sitting/standing/walking durations, sitting proportion, steps
- Heart rate: duration-weighted daily mean/std of HR ratio
- Noise: duration-weighted daily mean/std
- EMG: right-side daily p90/p50 (scaled to 0–1)

HR duration fallback: If watch times are missing, per-day durations fallback to the mean of available HR session durations.

Prepare Single-Instance Metrics (Metadata + IPAQ/OSPAQ)

```
from oh_stats import prepare_single_instance_metrics

# Single-instance metrics (metadata + baseline questionnaires)
single_ds = prepare_single_instance_metrics(profiles)
print(single_ds["data"].head())
```

Included metrics (when available):

- Metadata (all fields under meta_data)
- IPAQ: ordinal level (leve/moderada/alta → 1/2/3) + total_met
- OSPAQ: sitting percentage scaled to 0–1
- Weekly HAR total duration (summed across days)

---

10. Step 3: Check Data Quality

ALWAYS DO THIS!

Never skip data quality checks before modeling.

The Non-Negotiable Pre-Modeling Checks

```python
from oh_stats import summarize_outcomes, check_variance, missingness_report

# 1. Basic summary
summary = summarize_outcomes(ds)
print(summary)

# 2. Check for missing data
missing = missingness_report(ds)
high_missing = missing[missing['pct_missing'] > 10]
if len(high_missing) > 0:
    print(f"[WARNING] High missingness (>10%): {high_missing['outcome'].tolist()}")

# 3. Check for degenerate variables
variance = check_variance(ds)
degenerate = variance[variance['is_degenerate']]['outcome'].tolist()
if degenerate:
    print(f"[EXCLUDE] Cannot model: {degenerate}")
```

What to Look For

Check	Threshold	Action
Missing data	$> 10\%$	Investigate pattern; is it random or systematic?
Degenerate	mode $> 95\%$ of values	Exclude from modeling
Extreme skewness	$\lvert\text{skew}\rvert > 2$	Consider LOG transform
Sample size	< 20 subjects	Results may be unstable

---

11. Step 4: Fit Models

Single Outcome

```python
from oh_stats import fit_lmm

# Fit a Linear Mixed Model
result = fit_lmm(ds, "EMG_intensity.mean_percent_mvc")

# Check convergence
if result['converged']:
    print("Model fitted successfully!")
else:
    print("WARNING: Model had problems converging")
    print(result['warnings'])
```

Multiple Outcomes (Batch)

```python
from oh_stats import fit_all_outcomes

# Fit all outcomes
results = fit_all_outcomes(ds, skip_degenerate=True)

# Or limit to specific outcomes
results = fit_all_outcomes(
    ds,
    outcomes=["EMG_intensity.mean_percent_mvc", "EMG_apdf.active.p50"],
    max_outcomes=10
)
```

Model Options

```python
# Day as categorical (default) - tests each day vs Day 1
result = fit_lmm(ds, outcome, day_as_categorical=True)

# Day as linear trend - tests linear change per day
result = fit_lmm(ds, outcome, day_as_categorical=False)

# Apply transformation
from oh_stats import TransformType
result = fit_lmm(ds, outcome, transform=TransformType.LOG)

# Exclude side effect
result = fit_lmm(ds, outcome, include_side=False)
```

---

12. Step 5: Apply Multiplicity Correction

```python
from oh_stats import apply_fdr

# Apply FDR correction across outcomes
fdr_results = apply_fdr(results)
print(fdr_results)
```

Output:

```
                           outcome    p_raw  p_adjusted  significant
EMG_intensity.mean_percent_mvc       0.0003      0.0015         True
EMG_intensity.max_percent_mvc        0.0001      0.0015         True
EMG_apdf.active.p10                  0.0180      0.0360         True
EMG_apdf.active.p50                  0.0712      0.0712        False
```

---

13. Step 6: Post-Hoc Contrasts

> **⚠ Important**
>
> Only run post-hocs for outcomes that passed FDR correction!

```python
from oh_stats import pairwise_contrasts

# Get specific day comparisons
contrasts = pairwise_contrasts(result, "day_index", ds, adjustment="holm")
print(contrasts[["contrast", "estimate", "p_adjusted", "cohens_d"]])
```

Output:

```
    contrast  estimate  p_adjusted  cohens_d
0  Day1-Day2    -0.411       0.618    -0.059
1  Day1-Day3    -0.028       0.973    -0.004
2  Day1-Day4    -1.931       0.043    -0.276  <-- Significant!
3  Day1-Day5    -1.643       0.184    -0.235
```

---

14. Step 7: Check Diagnostics

```python
from oh_stats import residual_diagnostics

diag = residual_diagnostics(result)

print(f"Normality test p-value: {diag['normality_p']:.4f}")
print(f"Number of outliers: {diag['n_outliers']}")
print(f"Assumptions broadly met: {diag['assumptions_met']}")
```

PART III: Interpreting and Reporting Results

---

15. Understanding the Output

Coefficients Table

```
           term  estimate  std_error  z_value  p_value  ci_lower  ci_upper
      Intercept     9.406      1.035    9.087    0.000     7.377    11.434
C(day_index)[T.2]  -0.411      0.825   -0.498    0.618    -2.029     1.206
C(day_index)[T.3]  -0.028      0.839   -0.033    0.973    -1.672     1.616
C(day_index)[T.4]  -1.931      0.840   -2.298    0.022    -3.577    -0.284
C(day_index)[T.5]  -1.643      0.975   -1.685    0.092    -3.554     0.268
 C(side)[T.right]   0.902      0.550    1.641    0.101    -0.175     1.980
```

How to read this:

Column	What it means
term	What's being compared
estimate	The size of the difference (in raw units)
std_error	How uncertain we are (smaller = more confident)
z_value	Test statistic (estimate / std_error)
p_value	Probability this is just random chance
ci_lower/upper	95% confidence interval

Interpreting each row:

Row	Interpretation
Intercept (9.406)	Mean %MVC on Day 1, Left side
C(day_index)[T.2] = -0.411	Day 2 is 0.41 units LOWER than Day 1 (not significant)
C(day_index)[T.4] = -1.931	Day 4 is 1.93 units LOWER than Day 1 (p=0.02, **significant!**)
C(side)[T.right] = 0.902	Right side is 0.90 units HIGHER than left (not significant)

Random Effects

```python
print(result['random_effects'])
# {'group_var': 24.05, 'residual_var': 23.88, 'icc': 0.502}
```

Component	Value	Meaning
group_var	24.05	Between-subject variance (σ_u^2)
residual_var	23.88	Within-subject variance (σ^2)
icc	0.502	50% of variation is between subjects

Validation

ICC of 0.50 tells us: Mixed models were definitely the right choice! Half of all variation is just "who the person is."

Fit Statistics

```
print(result['fit_stats'])
# {'aic': 478.4, 'bic': 502.1, 'loglik': -234.2,
#  'lrt_stat': 12.5, 'lrt_df': 4, 'lrt_pvalue': 0.014}
```

Statistic	Use
AIC/BIC	Compare models (lower = better)
loglik	Log-likelihood (for advanced comparisons)
lrt_pvalue	**The p-value used for FDR correction**

---

16. Reporting Template

Methods Section

Example Methods Text

Daily EMG metrics were analyzed using linear mixed models with day as a fixed effect (categorical) and random intercepts for subjects to account for repeated measurements within individuals. Side (left/right) was included as a fixed effect. Models were fitted using maximum likelihood estimation via statsmodels (Python). Given the exploratory nature of the analysis across N=10 EMG outcomes, p-values were adjusted using the Benjamini-Hochberg procedure to control the false discovery rate at 5%. Post-hoc pairwise comparisons between days were corrected using the Holm method. Effect sizes were calculated as Cohen's d using the residual standard deviation as the denominator.

Results Section

Example Results Text

We analyzed 320 observations from 37 subjects over 5 monitoring days (mean 4.3 days per subject, range 3–5). The intraclass correlation was 0.50 (95% CI: 0.35–0.65), indicating that 50% of the variance in EMG intensity was attributable to between-subject differences, justifying the use of mixed models.

After FDR correction, 4 of 10 EMG outcomes showed significant day effects (all p_adj < 0.05). For mean %MVC specifically, the overall day effect was significant (LRT $\chi^2(4) = 12.5$, p = 0.014). Post-hoc comparisons (Holm-adjusted) revealed that Day 4 was significantly lower than Day 1 (Δ = -1.93 %MVC, 95% CI: -3.58 to -0.28, Cohen's d = 0.28, p_adj = 0.043), representing a small effect.

What to Report (Checklist)

Element	Example	Where
Sample size	"37 subjects, 320 observations"	Methods/Results
ICC	"ICC = 0.50"	Results
FDR method	"Benjamini-Hochberg"	Methods
Omnibus test	"LRT $\chi^2(4) = 12.5$, p = 0.014"	Results
Effect estimate	"Δ = -1.93 %MVC"	Results
95% CI	"95% CI: -3.58 to -0.28"	Results
Effect size	"Cohen's d = 0.28"	Results
Adjusted p-value	"p_adj = 0.043"	Results

PART IV: Edge Cases & Troubleshooting

---

17. Common Problems and Solutions

17.1 Missing Days / Unbalanced Data

The situation: Some subjects have 5 days of data, others have only 3.

Good News

LMMs handle this naturally! They use all available data and don't require balanced designs.

What to watch for:

- Is missingness random or systematic? (e.g., do subjects drop out because they're injured?)
- Very few observations per subject (< 3) may cause convergence issues

```
# Check missingness patterns
missing = missingness_report(ds)
print(missing[missing['pct_missing'] > 10])
```

17.2 Degenerate Outcomes (No Variance)

The situation: An outcome is nearly constant (e.g., 95% of values are zero).

The problem: No variance = nothing to model.

Solution: Exclude these outcomes from analysis.

```
variance = check_variance(ds)
degenerate = variance[variance['is_degenerate']]
print(f"Exclude: {list(degenerate['outcome'])}")
```

17.3 Convergence Failures

The situation: `result['converged'] = False`

Warning

The optimizer couldn't find a stable solution. Results are unreliable.

What to try:

1. **Simplify the model**: Remove interactions, use `side="average"`
2. **Check for degenerate outcomes**: Near-constant values cause problems
3. **Check sample size**: Need enough subjects (ideally 20+)
4. **Look at warnings**: `result['warnings']` often explains the issue

```python
if not result['converged']:
    print("Warnings:", result.get('warnings', []))
    # Try simpler model
    ds_simple = prepare_daily_emg(profiles, side="average")
    result = fit_lmm(ds_simple, outcome)
```

17.4 EMG Left/Right Correlation (side="both")

The situation: You kept both sides as separate rows, but left and right from the same subject-day are correlated.

The problem: A subject-only random intercept doesn't fully capture same-day correlations.

Three defensible strategies:

Strategy	Pros	Cons
`side="average"`	Simplest, no correlation issue	Loses side-specific information
Analyze sides separately	Clean interpretation	Doubles the number of tests
Keep `side="both"`	More power	Slight model misspecification

Recommendation

Start with `side="average"` for simplicity.

17.5 Skewed Distributions

The situation: Residuals are not normally distributed (e.g., right-skewed EMG data).

Don't panic! LMMs are fairly robust to moderate non-normality, especially with larger samples.

When to act:

- Severe skewness (> 2) with small samples
- Heavy ceiling/floor effects

Solutions:

```python
import numpy as np

# Log transform (for positive values, especially right-skewed)
ds['data']['log_outcome'] = np.log1p(ds['data'][outcome])

# Or specify in fit_lmm
from oh_stats import TransformType
result = fit_lmm(ds, outcome, transform=TransformType.LOG1P)
```

17.6 Outliers

The situation: A few extreme values are pulling the model.

How to identify:

```
diag = residual_diagnostics(result)
print(f"Outliers (|z| > 3): {diag['n_outliers']}")

# See which observations
import numpy as np
outlier_idx = np.abs(diag['standardized']) > 3
print(ds['data'][outlier_idx])
```

What to do:

1. **Investigate**: Are they data errors or real extreme values?
2. **Sensitivity analysis**: Run with and without outliers
3. **Report both**: "Results were similar with outliers excluded (N=2)"

17.7 Likert/Ordinal Data

The situation: You have questionnaire items on a 1-5 or 0-10 scale.

The theoretical issue: Likert scales are ordinal, not continuous. The difference between 1$→2*isn′tnecessarilythesameas*4→$5.

Practical guidance:

Distribution	Recommendation
Roughly symmetric, no ceiling/floor	Treat as continuous with LMM (common practice)
Heavy ceiling (most responses = max)	Consider ordinal models or dichotomize
Heavy floor (most responses = min)	Consider ordinal models or dichotomize

If treating as continuous: Always report medians and IQR alongside means.

17.8 Proportions (0-1 bounded)

The situation: Your outcome is a proportion (e.g., % time in a posture).

The problem: Values bounded at 0 and 1; residuals can't be normal at the extremes.

Solution: LOGIT transform

```
# Automatic via registry for registered proportions
result = fit_lmm(ds, 'ospaq_sitting_pct')  # Auto-applies LOGIT

# Or manual
result = fit_lmm(ds, outcome, transform=TransformType.LOGIT)
```

17.9 Small Sample Sizes

The situation: You have fewer than 30 subjects.

The problem: Random effect variance estimates become imprecise; model may not converge.

```
if result['n_groups'] < 30:
    print(f"Warning: Only {result['n_groups']} subjects")
    print("Consider: wider CIs, simpler models, sensitivity analyses")
```

Recommendations:

- Prefer simpler models (fewer fixed effects)
- Consider reporting alongside bootstrap CIs
- Be cautious about random effect variance interpretations

---

18. Quick Troubleshooting Checklist

Troubleshooting Checklist

☐ **Model didn't converge?**
→ Try side="average", check for degenerate outcomes, simplify model

☐ **Residuals look weird?**
→ Check for outliers, consider transformation

☐ **Unexpected results?**
→ Check missingness patterns, verify data quality

☐ **p-values all non-significant but you expected effects?**
→ Check ICC (high ICC = less power), check sample size

☐ **Too many significant results?**
→ Are you using FDR correction? Check for data leakage

PART V: Reference

---

19. Data Types and Transform Guide

When to Use Each Transform

Outcome Type	Transform	When to Use
Continuous (unbounded)	NONE	Default for %MVC, BPM, etc.
Right-skewed continuous	LOG or LOG1P	When distribution has long right tail
Proportions (0-1)	LOGIT	% time, rest_percent, OSPAQ
Counts	LOG1P	Number of events (pragmatic fallback)
Ordinal (5+ levels)	NONE	NPRS 0-10, ROSA 1-10 (treat as continuous)

Pre-Registered Outcomes

Outcome	Type	Transform
EMG_intensity.mean_percent_mvc	CONTINUOUS	NONE
EMG_intensity.iemg_percent_seconds	CONTINUOUS	LOG
EMG_apdf.rest_percent	PROPORTION	LOGIT
EMG_muscular_rest.gap_count	COUNT	LOG1P
copsoq_*	CONTINUOUS	NONE
mueq_*	CONTINUOUS	NONE
rosa_total	ORDINAL	NONE
ipaq_met_min_week	CONTINUOUS	LOG1P
ospaq_*_pct	PROPORTION	LOGIT
nprs_*	ORDINAL	NONE

---

20. Glossary

Term	Definition
AIC	Akaike Information Criterion. Lower = better model fit.
AnalysisDataset	Standardized container for analysis-ready data.

Term	Definition
Coefficient	Estimated size of an effect (e.g., Day 4 is -1.93 lower than Day 1).
Cohen's d	Standardized effect size: difference / standard deviation.
Confidence Interval (CI)	Range that probably contains the true effect. 95% CI means 95% confident.
Converged	Model successfully found a solution. If FALSE, results unreliable.
FDR	False Discovery Rate. Controls expected proportion of false positives.
Fixed Effect	Something we're interested in measuring (e.g., day effect, side effect).
FWER	Family-Wise Error Rate. Controls chance of ANY false positive.
ICC	Intraclass Correlation. Proportion of variance due to between-subject differences.
LMM	Linear Mixed Model. Handles repeated measures via fixed + random effects.
LRT	Likelihood Ratio Test. Compares nested models to test if a factor matters.
p-value	Probability of seeing your data if there were no real effect.
Random Effect	Variation we account for but don't directly measure (e.g., subject baselines).
Residual	The "leftover" after the model's prediction.
Transform	Converting data (e.g., LOG) to make it better behaved for modeling.

---

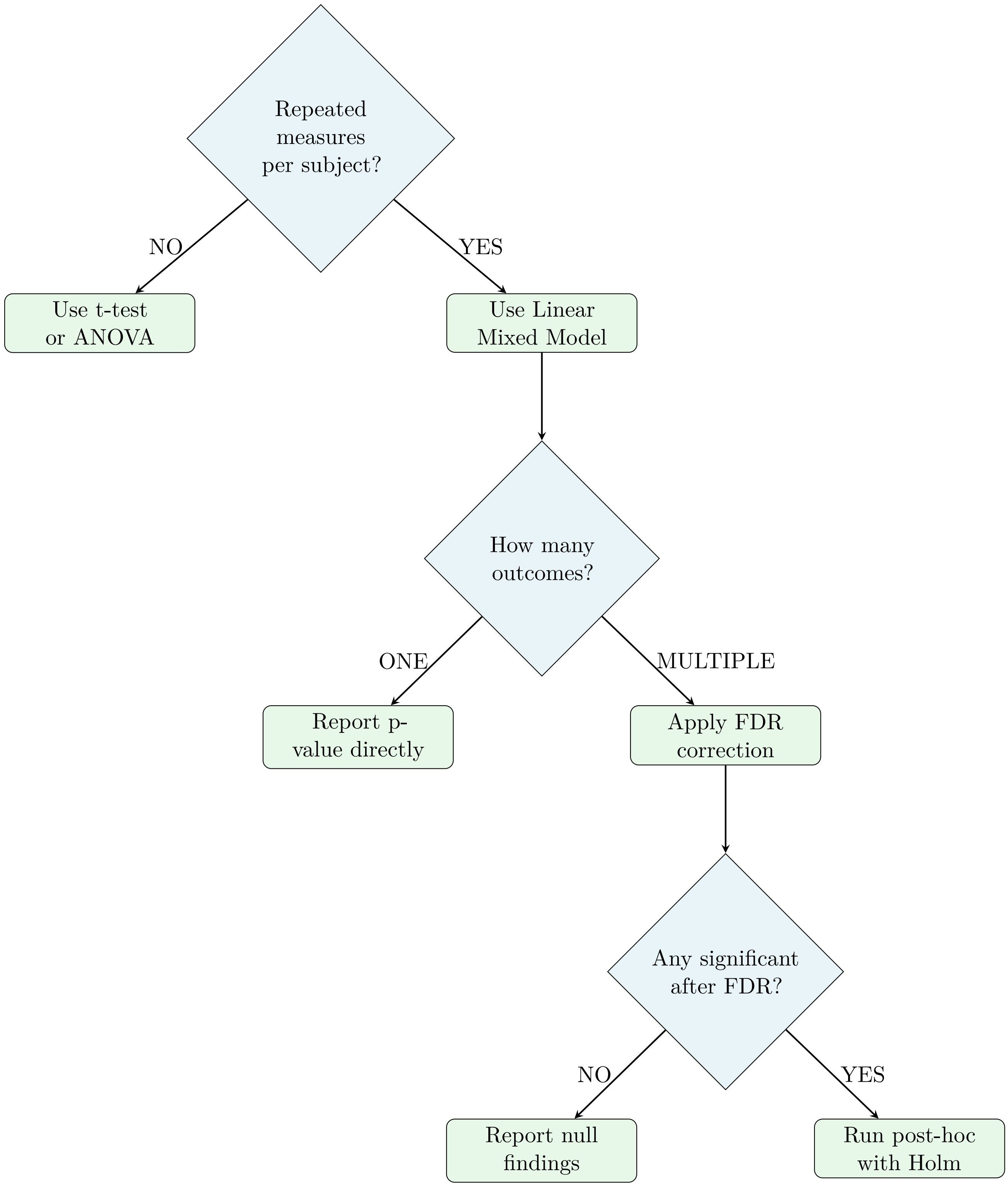
21. Quick Reference Card

Minimal Workflow

```
from oh_parser import load_profiles
from oh_stats import (
    get_profile_summary,
    prepare_daily_emg,
    prepare_daily_metrics,
    prepare_single_instance_metrics,
    summarize_outcomes,
    check_variance,
    fit_all_outcomes,
    apply_fdr,
```

```python
)

# 1. Load & Discover
profiles = load_profiles("/path/to/data")
print(get_profile_summary(profiles))

# 2. Prepare
ds = prepare_daily_emg(profiles, side="average")

# Or unified daily metrics
daily_ds = prepare_daily_metrics(profiles)

# Single-instance metrics
single_ds = prepare_single_instance_metrics(profiles)

# 3. Check Quality
print(summarize_outcomes(ds))
print(check_variance(ds))

# 4. Model
results = fit_all_outcomes(ds, skip_degenerate=True)

# 5. Correct
fdr = apply_fdr(results)

# 6. Report
print(fdr[fdr['significant']])
```

Decision Tree


Repeated measures per subject?
NO
YES
Use t-test or ANOVA
Use Linear Mixed Model
How many outcomes?
ONE
MULTIPLE
Report p-value directly
Apply FDR correction
Any significant after FDR?
NO
YES
Report null findings
Run post-hoc with Holm

OH Stats Complete Guide v1.0

January 2026

For oh_stats package v0.3.0