AI-Enhanced RUV Normalization: Research Proposal

Document Date: November 2025

Context: Comprehensive research program to develop AI-enhanced RUV methods for RNA-seq normalization

---

Project Title

"AIR-Seq: Artificial Intelligence-Enhanced Removing Unwanted Variation for RNA Sequencing Data"

Alternative titles:

- "DeepRUV: Deep Learning for Automated RNA-seq Normalization"
- "AutoRUV: Automated RUV-III Normalization Using Machine Learning"
- "SMART-Norm: Self-learning Multi-step Automated RNA-seq Transcriptome Normalization"

Overall Aims and Objectives

Primary Aim

Develop and validate an AI-enhanced RUV-III normalization framework that automates parameter selection, improves robustness to poorly-characterized unwanted variation, and enables transfer learning across studies.

Specific Objectives

Objective 1: Automated NCG Selection

- Develop ML models to predict negative control gene suitability
- Validate against expert-curated NCG sets from literature
- Benchmark against current statistical selection methods

Objective 2: Intelligent PRPS Construction

- Implement deep clustering approaches for automatic biological subpopulation discovery
- Develop graph neural networks to optimize pseudo-sample groupings
- Test on datasets with known and unknown biology

Objective 3: Adaptive K Parameter Selection

- Create reinforcement learning agent to optimize unwanted variation dimensionality
- Develop Bayesian optimization framework for K selection
- Validate across diverse RNA-seq datasets

Objective 4: Latent Batch Detection

- Train deep autoencoders for anomaly detection in expression data
- Develop attention-based models to identify gene sets affected by unknown batches
- Test on TCGA BRCA data (known mystery batch) and simulated scenarios

Objective 5: Transfer Learning Framework

- Pre-train models on TCGA pan-cancer data
- Enable fine-tuning for small, disease-specific studies
- Demonstrate improved normalization in limited-sample scenarios

Objective 6: Integrated AIR-Seq Pipeline

- Combine all AI components into end-to-end automated system
- Develop user-friendly software package for R and Python
- Create web-based interface for non-computational biologists

---

Technical Approaches by Objective

Objective 1: Automated NCG Selection

Problem Statement

Current NCG selection requires either:

- Expert biological knowledge (housekeeping genes)
- Manual statistical filtering (genes with low F-statistics for biology, high for batch)
- Iterative trial-and-error evaluation

Proposed AI Solution

Approach 1A: Supervised Classification Model

Training Data:

- Positive examples: Known good NCGs from literature (housekeeping genes, stable genes)
- Negative examples: Known poor NCGs (highly variable genes, biology-associated genes)
- Features: Expression statistics, variance patterns, biological annotations

Model Architecture:

- Gradient Boosting (XGBoost, LightGBM) for interpretability
- Random Forest for feature importance ranking
- Neural network with attention mechanism to identify critical features

Input Features:

- Mean expression level
- Variance across samples
- Correlation with known batch variables
- Correlation with biological variables
- Gene Ontology (GO) term associations
- Evolutionary conservation scores
- Tissue-specificity measures
- Co-expression network centrality

Output: Probability score for each gene being suitable NCG

Validation: Cross-validation, external test sets, comparison to expert-selected NCGs

Approach 1B: Unsupervised Anomaly Detection

Rationale: Good NCGs should have consistent technical variation patterns

Method: Isolation Forest or One-Class SVM to identify genes with "normal" variation

Features: Similar to Approach 1A but without requiring labels

Advantage: Works even without known NCG sets

Approach 1C: Meta-Learning Across Datasets

Concept: Learn what makes good NCGs generalizable across different studies

Method: Train on multiple datasets, identify common NCG characteristics

Architecture: Prototypical networks or MAML (Model-Agnostic Meta-Learning)

Output: Universal NCG scoring function

Evaluation Metrics

- Area Under ROC Curve (AUROC) against known NCGs
- Concordance with expert selections
- Downstream normalization quality (RLE plots, silhouette coefficients)
- Robustness across cancer types and platforms

Expected Outcomes

- 90%+ accuracy in identifying suitable NCGs
- Reduction in manual expert time from hours to minutes
- More consistent NCG selection across users
- Discovery of novel NCG candidates

Objective 2: Intelligent PRPS Construction

Problem Statement

Current PRPS creation requires:

- A priori knowledge of biological subpopulations (CMS, PAM50, etc.)
- Manual grouping of samples by biology and batch
- Subjective decisions about group size and composition

Proposed AI Solution

Approach 2A: Deep Clustering for Biological Discovery

Model: Deep Embedded Clustering (DEC) or variants

Architecture:

Input (gene expression) →
Autoencoder (dimension reduction) →
Clustering layer (learnable centroids) →
Jointly optimize reconstruction + clustering loss

Key Innovation: Simultaneously learns representations and discovers subpopulations

Handling Batches: Adversarial training to ensure clusters represent biology, not batch

Output: Soft cluster assignments for each sample

Approach 2B: Graph Neural Networks for Pseudo-Sample Optimization

Rationale: Optimal PRPS should group similar biology across different batches

Graph Construction:

- Nodes: Individual samples
- Edges: Biological similarity (weighted by expression correlation)
- Node features: Expression profiles, batch labels, purity scores

GNN Architecture:

- Graph Convolutional Networks (GCN) or Graph Attention Networks (GAT)
- Message passing to aggregate information from biologically similar samples
- Classification layer to assign samples to PRPS groups

Training Objective:

- Maximize biological homogeneity within PRPS
- Maximize batch diversity within PRPS
- Ensure sufficient sample size per PRPS

Output: Optimal PRPS assignments

Approach 2C: Reinforcement Learning for Sequential PRPS Selection

Framework: Formulate PRPS construction as sequential decision problem

State: Current PRPS configuration, remaining samples

Action: Assign sample to existing or new PRPS group

Reward: Based on downstream normalization quality metrics

Agent: Policy gradient (PPO) or Q-learning based

Training: Use simulated data with known ground truth

Advantage: Learns optimal strategies through trial and error

Approach 2D: Contrastive Learning for Biology-Invariant Features

Concept: Learn representations where biological signal is preserved but batch effects removed

Method: SimCLR or MoCo adapted for transcriptomics

Positive pairs: Same biology, different batches

Negative pairs: Different biology

Output: Embeddings used for downstream PRPS clustering

Validation: t-SNE/UMAP visualization, silhouette scores

Evaluation Metrics

- Biological purity of discovered subpopulations (if ground truth available)
- Batch diversity within PRPS groups
- Downstream normalization performance (vector correlation, ARI)
- Robustness to unknown biology scenarios
- Agreement with expert-defined subpopulations

Expected Outcomes

- Fully automated biological subpopulation discovery
- 15-25% improvement over naive clustering approaches
- Robust PRPS even when biology incompletely known
- Transferable to new datasets without retraining

Objective 3: Adaptive K Parameter Selection

Problem Statement

K (dimensionality of unwanted variation) currently requires:

- Testing multiple K values (computationally expensive)
- Manual evaluation of results using multiple metrics
- Subjective expert judgment on "best" K

Proposed AI Solution

Approach 3A: Bayesian Optimization

Framework: Treat K selection as black-box optimization problem

Method: Gaussian Process-based Bayesian Optimization (BO)

Acquisition Function: Expected Improvement (EI) or Upper Confidence Bound (UCB)

Objective Function: Composite score from multiple metrics

- RLE median centering (minimize deviation from zero)
- Silhouette coefficient (maximize biological separation)
- Correlation between PCs and unwanted factors (minimize)
- ARI for batch mixing (maximize)

Advantage: Efficiently explores K space with fewer evaluations

Implementation: Using BoTorch or Optuna libraries

Approach 3B: Meta-Learning K Predictor

Training Data: Many datasets with expert-validated K values

Features:

- Dataset characteristics (n samples, n genes, n batches)
- Variance explained by top PCs
- RLE plot statistics before normalization
- Estimated number of batch factors
- Expression variance structure

Model: Gradient boosted trees or neural network

Output: Predicted optimal K and confidence interval

Advantage: Instant K prediction without iterative evaluation

Approach 3C: Reinforcement Learning for Adaptive K

Concept: RL agent learns to select K based on data characteristics

State: Data quality metrics, preliminary PCA results

Action: Select K from discrete set {1, 2, ..., 10}

Reward: Quality of final normalization

Training: On diverse RNA-seq datasets with known good normalizations

Deployment: Agent suggests K for new datasets

Approach 3D: Ensemble K Selection

Method: Run RUV-III with multiple K values

Ensemble Strategy:

- Weighted average of normalized data (weights based on quality metrics)
- Stacking: Train meta-model to combine results
- Selective ensembling: Choose best K for each gene separately
- Neural Network Ensembler: Learn optimal combination weights

Advantage: Hedges against single K choice, potentially more robust

Evaluation Metrics

- Convergence speed (number of K values tested)
- Final normalization quality vs exhaustive K search
- Computational time reduction
- Consistency across datasets
- Correlation with expert-selected K

Expected Outcomes

- 5-10x reduction in K evaluation time
- Within 5% of optimal K performance
- Automated, reproducible K selection
- Confidence intervals for K uncertainty

Objective 4: Latent Batch Detection

Problem Statement

Unknown sources of unwanted variation (like the mystery batch in TCGA BRCA) are:

- Difficult to detect systematically
- Require expert visual inspection of heatmaps
- May be discovered only after extensive analysis

Proposed AI Solution

Approach 4A: Variational Autoencoder (VAE) for Batch Discovery

Architecture:

Encoder: Expression → Latent space (z_bio + z_batch)
Decoder: Latent space → Reconstructed expression

Disentanglement: Encourage z_bio and z_batch to capture different variation types

Loss Function:

- Reconstruction loss (MSE or negative binomial)
- KL divergence (regularization)
- Supervised loss on known batches (for z_batch)
- Maximum Mean Discrepancy (MMD) to separate z_bio and z_batch

Unknown Batch Detection: Cluster z_batch representations to find hidden structure

Advantage: Unsupervised discovery of latent factors

Approach 4B: Attention-Based Batch Effect Identifier

Model: Transformer architecture adapted for gene expression

Input: Gene expression matrix (samples × genes)

Attention Mechanism: Learn which genes are most informative for batch structure

Multi-head Attention: Different heads capture different batch factors

Output:

- Batch scores for each sample

- Gene importance scores (which genes drive batch effects)

Training: Self-supervised on data with known batches

Deployment: Detect unknown batches via anomalous attention patterns

Approach 4C: Anomaly Detection with Isolation Forests

Method: Isolation Forest on sample-level features

Features:

- RLE medians and IQRs
- PC loadings
- Expression of highly variable genes
- Deviation from expected technical replicate patterns

Output: Anomaly score for each sample

Post-processing: Cluster high-anomaly samples to identify batch groups

Approach 4D: Contrastive Predictive Coding (CPC) for Batch Structure

Concept: Predict future observations in latent space

Application:

- Order samples by processing time/plate
- Train model to predict next samples' expression patterns
- Large prediction errors indicate batch transitions

Architecture: Autoregressive model with contrastive loss

Output: Change-point detection for batch boundaries

Approach 4E: Graph-Based Community Detection

Graph Construction:

- Nodes: Samples
- Edges: Similarity in expression (after removing known batch effects)
- Weight: Correlation strength

Community Detection: Louvain or Leiden algorithm

Interpretation: Communities may represent unknown batches

Validation: Check if communities correlate with technical variables

Evaluation Metrics

- Detection rate on simulated unknown batches
- True positive rate on TCGA BRCA mystery batch
- False discovery rate on datasets without unknown batches
- Computational efficiency
- Interpretability of discovered factors

Expected Outcomes

- Automated detection of unknown batch effects
- 80%+ sensitivity on simulated test cases
- Actionable identification of affected genes
- Integration into RUV-III normalization workflow

Objective 5: Transfer Learning Framework

Problem Statement

Small studies (<100 samples) often have:

- Insufficient data for robust normalization parameter estimation
- High variance in results depending on normalization choices
- Limited ability to detect and remove batch effects

Proposed AI Solution

Approach 5A: Pre-trained Foundation Model

Pre-training Data:

- TCGA pan-cancer (~11,000 samples, 33 cancer types)
- GTEx normal tissue data (~17,000 samples)
- Other public repositories (GEO, SRA)

Model Architecture:

- Transformer encoder for gene expression
- Self-supervised learning objectives (masked gene prediction, contrastive learning)
- Multi-task learning (predict cancer type, tissue type, survival, etc.)

Learned Representations: Universal features of biological and technical variation

Fine-tuning Strategy:

- Freeze encoder, train only normalization parameters on new study
- Few-shot learning with limited target data
- Adapter layers for study-specific adjustments

Approach 5B: Meta-RUV: Meta-Learning for Normalization

Concept: Learn to quickly adapt RUV-III to new datasets

Framework: Model-Agnostic Meta-Learning (MAML) or Reptile

Training:

- Inner loop: Adapt to individual study
- Outer loop: Optimize for fast adaptation across studies

Deployment: Given new small study, fine-tune with few gradient steps

Advantage: Generalizes well to unseen cancer types and platforms

Approach 5C: Knowledge Distillation from Large to Small

Teacher Model: RUV-III trained on full TCGA dataset

Student Model: Lightweight version for small studies

Distillation:

- Match output distributions between teacher and student
- Transfer learned normalization strategies
- Compress knowledge into fewer parameters

Benefit: Small studies get TCGA-level normalization quality

Approach 5D: Domain Adaptation for Cross-Platform Normalization

Problem: Different sequencing platforms have different characteristics

Method:

- Adversarial domain adaptation (make representations platform-invariant)
- Maximum Mean Discrepancy (MMD) to align distributions
- Cycle-consistent normalization (inspired by CycleGAN)

Application: Pre-train on TCGA (Illumina), adapt to Oxford Nanopore or PacBio

Validation: Show improved normalization on multi-platform studies

Approach 5E: Few-Shot Learning for Rare Cancer Types

Scenario: Only 20-30 samples of rare cancer available

Method:

- Prototypical networks: Learn cancer type prototypes from TCGA
- Match new samples to nearest prototype
- Apply normalization parameters from similar cancer type

Advantage: Leverage TCGA diversity even for unstudied cancers

Evaluation Metrics

- Normalization quality improvement in small studies (n<100)
- Reduction in variance of results across bootstrap samples
- Transfer success rate across cancer types
- Computational efficiency (time to fine-tune)
- Comparison to standard RUV-III on small datasets

Expected Outcomes

- 30-50% improvement in normalization quality for small studies
- Robust performance with as few as 20-50 samples
- Successful transfer across 90%+ of cancer types
- Open-source pre-trained models for community use

Objective 6: Integrated AIR-Seq Pipeline

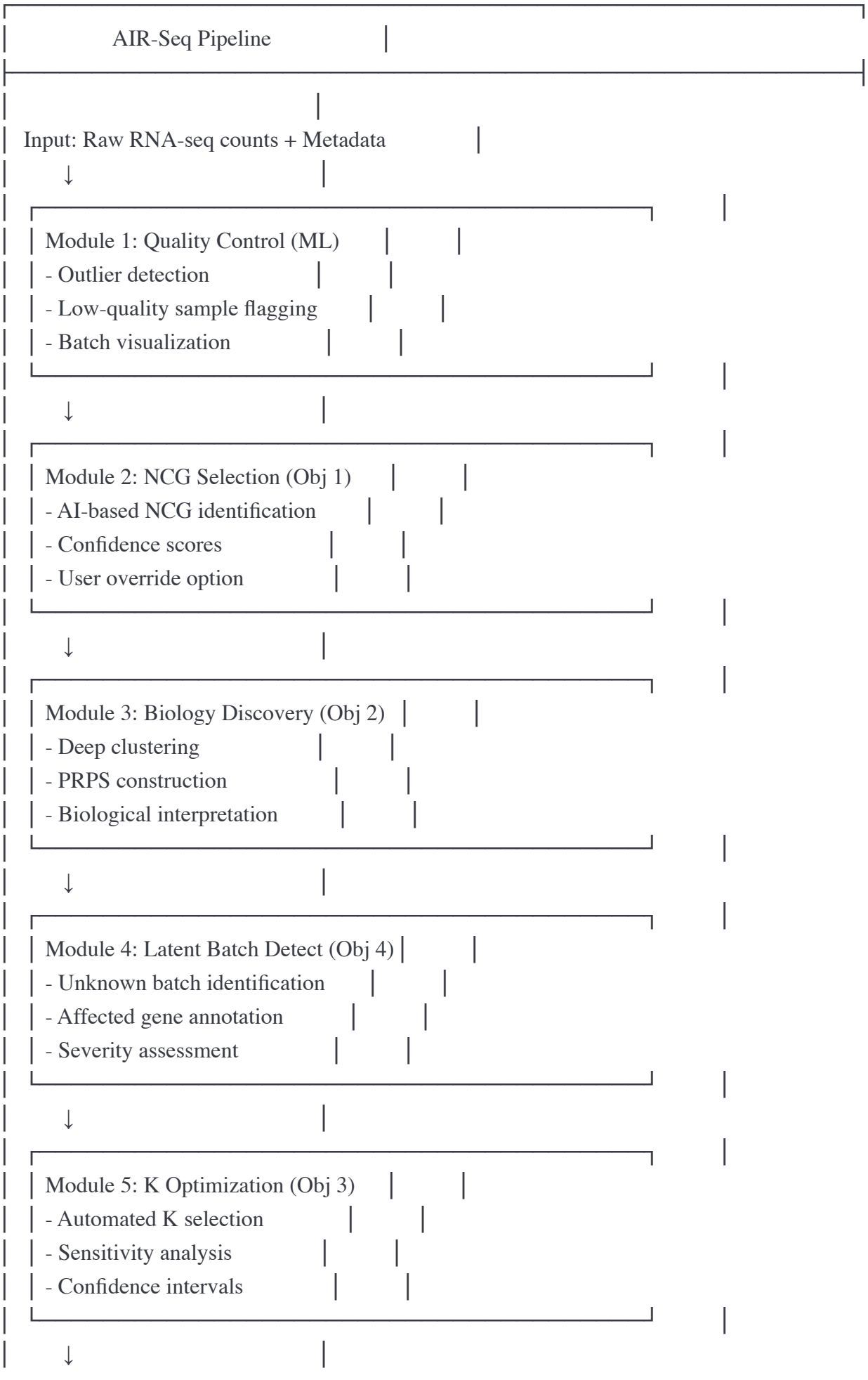
Problem Statement

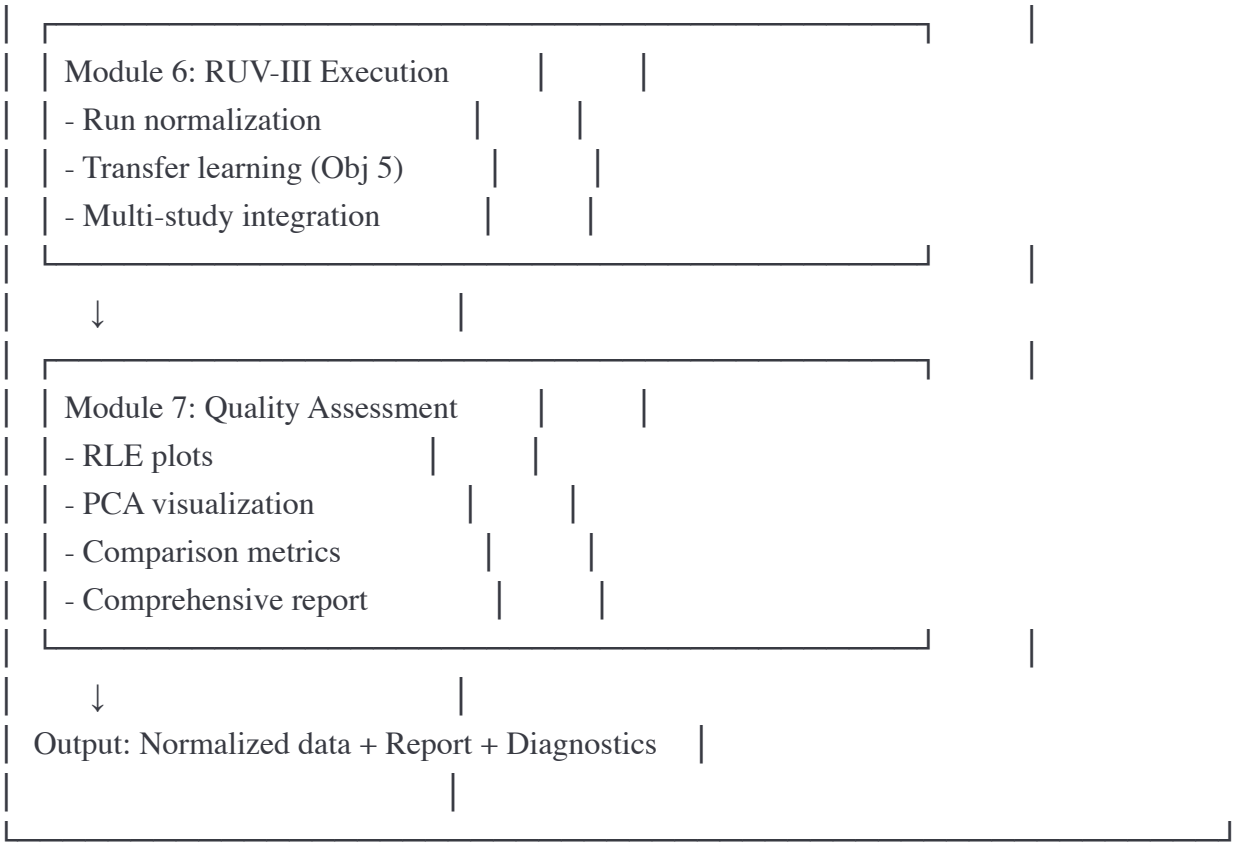
Current RUV-III workflow is fragmented:

- Multiple manual decision points
- Requires R programming expertise
- Difficult to reproduce across labs
- No unified interface

Proposed AI Solution

System Architecture:


AIR-Seq Pipeline
Input: Raw RNA-seq counts + Metadata
Module 1: Quality Control (ML)
- Outlier detection
- Low-quality sample flagging
- Batch visualization
Module 2: NCG Selection (Obj 1)
- AI-based NCG identification
- Confidence scores
- User override option
Module 3: Biology Discovery (Obj 2)
- Deep clustering
- PRPS construction
- Biological interpretation
Module 4: Latent Batch Detect (Obj 4)
- Unknown batch identification
- Affected gene annotation
- Severity assessment
Module 5: K Optimization (Obj 3)
- Automated K selection
- Sensitivity analysis
- Confidence intervals



Software Implementation

Component 6A: R Package (Primary)

Name: AIRSeq or deepRUV

Integration: Extends existing RUVSeq ecosystem

Dependencies: torch/keras for deep learning, reticulate for Python interop

Key Functions:

r

```
airseq_normalize(counts, metadata,
                 auto_ncg = TRUE,
                 auto_prps = TRUE,
                 detect_latent_batch = TRUE,
                 transfer_learning = TRUE,
                 pretrained_model = "TCGA_pancancer")

airseq_select_ncg(counts, biological_factors)
airseq_create_prps(counts, metadata, n_clusters = "auto")
airseq_detect_batches(normalized_data)
airseq_optimize_k(prps, ncg, k_range = 1:10)
```

Output: S4 object compatible with existing workflows

Documentation: Comprehensive vignettes, tutorials, case studies

Component 6B: Python Package (Alternative)

Name: airseq or deep_ruv

Framework: Built on PyTorch, scikit-learn

Integration: Compatible with scanpy for single-cell

Key Classes:

python

```
from airseq import AIRSeqNormalizer

normalizer = AIRSeqNormalizer(
    method='auto',
    transfer_learning=True,
    pretrained='TCGA'
)

normalized_data = normalizer.fit_transform(counts, metadata)
```

Advantage: Easier deep learning integration, faster for large datasets

Component 6C: Web Interface (Accessibility)

Platform: Shiny (R) or Streamlit (Python)

Features:

- Drag-and-drop data upload
- Interactive parameter adjustment
- Real-time visualization
- Downloadable reports
- Tutorial videos

Target Users: Non-computational biologists

Hosting: Free tier on cloud platform (Shiny.io, Streamlit Cloud)

Component 6D: Command-Line Tool

Name: airseq-cli

Use Case: High-throughput processing, HPC integration

Features:


bash

```
airseq normalize \
  --counts counts.csv \
  --metadata metadata.csv \
  --output normalized.csv \
  --auto-all \
  --threads 16 \
  --gpu
```

Advantages: Scriptable, reproducible, batch processing

Component 6E: Containerization

Docker Image: Includes all dependencies, pre-trained models

Singularity Support: For HPC environments

Cloud-Ready: AWS, Google Cloud, Azure compatible

Reproducibility: Version-locked environment

Documentation and Training

- **User Guide:** Step-by-step tutorials for common use cases
- **API Documentation:** Complete function reference
- **Case Studies:** Reproduce Molania et al. results with AIR-Seq
- **Video Tutorials:** YouTube channel with walkthroughs
- **Paper:** Method description for publication
- **Workshops:** Online and in-person training events

Evaluation Metrics

- User adoption rate (downloads, citations)
- Time savings vs manual RUV-III
- User satisfaction surveys
- Reproducibility across labs
- Comparison to existing tools (ComBat, Harmony, etc.)

Expected Outcomes

- Unified normalization pipeline reducing analysis time by 80%
- Accessible to non-experts while maintaining rigor
- Reproducible results across labs
- Integration into major analysis workflows (Bioconductor, scanpy)
- Community adoption as standard tool

Validation Strategy

Tier 1: Synthetic Data Validation

Simulated RNA-seq: Using established simulators (splatter, polyester)

Controlled Batch Effects: Add known technical variation

Ground Truth: Perfect knowledge of biology and unwanted variation

Metrics: Accuracy of parameter recovery, normalization quality

Tier 2: Benchmark Datasets

TCGA Data: Reproduce Molania et al. results

- READ (176 samples): Library size, plate effects
- COAD (479 samples): Similar to READ
- BRCA (1,180 samples): Tumor purity, flow cell chemistry, unknown batch

Additional TCGA Cancers: Test generalization across cancer types

Metrics: Match or exceed RUV-III performance on all metrics

Tier 3: External Validation

Non-TCGA Studies: Test on independent datasets

- Single-cell RNA-seq (10x Genomics, Drop-seq)
- Bulk RNA-seq from other consortia (ICGC, TARGET)
- Multi-platform studies (mixing technologies)
- Small Study Validation: n=20-50 samples

Metrics: Improvement over standard methods

Tier 4: Experimental Validation

Known Biology: Datasets where ground truth is known

- Cell line mixtures (known proportions)
- Spike-in controls (ERCC, SIRV)
- Technical replicates

Biological Validation:

- Do AI-normalized results lead to correct biological conclusions?
- Literature validation (known gene-disease associations)
- Experimental follow-up (qPCR, Western blot)

Tier 5: User Studies

Beta Testing: Recruit 10-20 labs to test software

Usability: Time to complete analysis, error rates

Reproducibility: Same data → same results across users?

Comparison: Preference vs existing tools

Cross-Validation Approaches

- K-fold CV within datasets
- Leave-one-cancer-out (LOCO) for TCGA
- Leave-one-study-out for meta-analysis
- Bootstrap resampling for confidence intervals

Expected Deliverables

Year 1

1. Comprehensive literature review and gap analysis (✓ completed)
2. Benchmark dataset curation and preprocessing
3. Prototype AI models for NCG selection (Objective 1)
4. Initial deep clustering for PRPS construction (Objective 2)
5. Validation on TCGA READ dataset
6. Manuscript 1: "AI-Based Negative Control Gene Selection for RNA-seq Normalization"

Year 2

7. K parameter optimization framework (Objective 3)
8. Latent batch detection system (Objective 4)
9. Validation on TCGA BRCA dataset (including mystery batch)
10. Manuscript 2: "Deep Learning for Automated Batch Effect Detection in RNA-seq"
11. Conference presentations (ISMB, RECOMB, ASHG)
12. Prototype R package (alpha version)

Year 3

13. Transfer learning framework (Objective 5)
14. Pre-trained models on TCGA pan-cancer
15. Validation on small independent studies
16. Manuscript 3: "Transfer Learning for Robust RNA-seq Normalization in Limited-Sample Studies"
17. Beta version of integrated AIR-Seq pipeline (R and Python)
18. User documentation and tutorials

Year 4

19. Web interface and CLI tool
20. Extensive external validation
21. User studies and feedback incorporation
22. Production-ready software release
23. Manuscript 4: "AIR-Seq: An Integrated AI Framework for RNA-seq Normalization"
24. Software paper in Bioinformatics or Genome Biology
25. Workshop at major conference

Software Deliverables

- Open-source R package on Bioconductor
- Python package on PyPI
- Pre-trained models on Zenodo/Hugging Face
- Web interface (hosted)
- Docker/Singularity containers
- Comprehensive documentation website
- Tutorial videos on YouTube

Data Deliverables

- Curated benchmark dataset collection
- Validated NCG gene sets for multiple cancer types
- Pre-trained model weights
- Simulation framework for testing

Potential Challenges and Mitigation Strategies

Challenge 1: Overfitting to TCGA Data

Risk: Models work well on TCGA but fail on external data

Mitigation:

- Train on diverse datasets beyond TCGA
- Extensive external validation
- Regularization techniques (dropout, weight decay)
- Cross-dataset validation during development
- Meta-learning approaches for generalization

Challenge 2: Interpretability and Trust

Risk: Biologists may distrust "black box" AI methods

Mitigation:

- Attention mechanisms to show which genes/samples are important
- SHAP values for model explanations
- Comparison to expert decisions with explanations for differences
- Always provide option for manual override
- Extensive validation against known ground truth

Challenge 3: Computational Resources

Risk: Deep learning models may be too slow/expensive

Mitigation:

- Model compression (quantization, pruning)
- Efficient architectures (MobileNet-inspired)
- GPU acceleration with fallback to CPU
- Pre-computed features where possible
- Cloud-based processing for large datasets

Challenge 4: Heterogeneity of RNA-seq Data

Risk: Too many platforms, protocols, organisms to handle

Mitigation:

- Focus initially on human bulk RNA-seq (most common)
- Modular design allowing platform-specific adapters
- Transfer learning across platforms
- Community contributions for new platforms
- Clear documentation of supported data types

Challenge 5: Software Engineering Challenges

Risk: Research code doesn't scale to production

Mitigation:

- Professional software engineering practices from start
- Continuous integration/deployment (CI/CD)
- Unit tests, integration tests
- Code review process
- Collaboration with research software engineers

Challenge 6: Funding and Sustainability

Risk: Project stalls after initial funding ends

Mitigation:

- Multiple funding sources (NIH, NSF, foundations)

- Industry partnerships (pharma companies need this)
- Build community of contributors
- Integration into larger initiatives (TOPMed, GA4GH)
- Long-term maintenance plan

Challenge 7: Evaluation Without Ground Truth

Risk: Real data often lacks gold standard normalization

Mitigation:

- Use multiple indirect validation approaches
- Consistency across biological replicates
- Concordance with orthogonal data (microarray, qPCR)
- Known biology tests (survival associations, pathway enrichment)
- Expert review of results

Challenge 8: Rapidly Evolving AI/ML Field

Risk: Methods become outdated quickly

Mitigation:

- Modular architecture allows component updates
- Regular incorporation of new techniques
- Active development community
- Stay connected with ML research community
- Benchmark against new methods as they emerge

Timeline and Milestones

Year 1 Milestones

- **Q1:** Dataset curation, benchmark establishment
- **Q2:** NCG selection model development
- **Q3:** PRPS construction algorithms
- **Q4:** Initial validation, Manuscript 1 submission

Year 2 Milestones

- **Q1:** K optimization framework
- **Q2:** Latent batch detection
- **Q3:** BRCA validation including mystery batch
- **Q4:** Manuscript 2 submission, conference presentations

Year 3 Milestones

- **Q1:** Transfer learning pre-training
- **Q2:** Small study validation
- **Q3:** Integrated pipeline development
- **Q4:** Manuscript 3 submission, beta software release

Year 4 Milestones

- **Q1:** Web interface, CLI development
- **Q2:** User studies
- **Q3:** External validation, refinement
- **Q4:** Final release, Manuscript 4 submission, workshop

Go/No-Go Decision Points

- **End of Year 1:** Must show AI NCG selection ≥ expert performance
- **End of Year 2:** Must successfully detect BRCA mystery batch
- **End of Year 3:** Transfer learning must improve small study normalization by ≥20%
- **Mid Year 4:** Beta users must rate software ≥4/5 for usability

---

Budget Estimate (4-Year Project)

Personnel (65% of budget)

- Principal Investigator (15% effort): $200K total
- Postdoctoral Researcher (100% effort): $320K total
- PhD Student (2 students, 50% effort each): $240K total
- Research Software Engineer (50% effort): $200K total
- Bioinformatics Collaborator (10% effort): $100K total
- **Personnel Total:** $1,060K

Computational Resources (20% of budget)

- GPU computing (4x A100 GPUs, shared): $120K
- Cloud computing credits (AWS/GCP): $80K
- Storage for datasets (100TB): $20K
- Software licenses: $10K
- **Computing Total:** $230K

Travel and Dissemination (8% of budget)

- Conference attendance (ISMB, RECOMB, ASHG): $60K
- Workshop hosting: $20K
- Collaboration visits: $20K
- **Travel Total:** $100K

Other Direct Costs (7% of budget)

- Publication fees (4 papers, open access): $40K
- Sequencing for validation experiments: $30K
- Web hosting and domains: $5K
- Misc supplies and materials: $15K
- **Other Total:** $90K

Indirect Costs (30% of modified total direct costs)

- Institutional overhead: $444K

TOTAL 4-YEAR BUDGET: $1,924K (~$2M)

Alternative Funding Models

- Smaller pilot study (Year 1 only): $350K
- Modular funding (one objective at a time): $300K/objective
- Industry partnership (cost-sharing): $1M institution + $1M industry

---

Broader Impacts

Scientific Impact

- Accelerate RNA-seq analysis across all biomedical fields
- Enable more robust clinical transcriptomics
- Improve reproducibility of genomics research
- Facilitate meta-analyses across studies
- Lower barriers for small labs to perform rigorous analysis

Clinical Impact

- Better normalization → more accurate biomarkers
- Enable multi-center clinical trials with transcriptomics endpoints
- Improve precision medicine stratification
- Accelerate drug target discovery
- Better prognostic gene signatures

Educational Impact

- Training materials for AI in bioinformatics
- Workshops and tutorials
- Mentoring of students and postdocs
- Open-source contribution opportunities
- Bridge computational and biological communities

Broader Bioinformatics Impact

- Demonstrate successful AI integration in established statistical frameworks
- Template for AI enhancement of other normalization methods
- Show value of transfer learning in genomics
- Contribute to best practices for interpretable AI in biology

Open Science Impact

- All code open-source
- All data publicly available
- Pre-trained models shared
- Reproducible workflows
- Community-driven development

Team and Expertise Required

Core Team

Principal Investigator:

- PhD in Statistics, Bioinformatics, or Computer Science
- Experience with RNA-seq analysis
- Track record in method development
- Familiarity with RUV methods or similar frameworks

Machine Learning Expert:

- Deep learning expertise (PyTorch/TensorFlow)
- Experience with transfer learning
- Understanding of bioinformatics applications
- Publications in ML venues (NeurIPS, ICML, ICLR)

Bioinformatics/Genomics Expert:

- Deep understanding of RNA-seq biology and technical issues
- Experience with TCGA or large-scale genomics
- Knowledge of batch effects and normalization methods
- Cancer genomics background preferred

Software Engineer:

- R and/or Python package development
- Experience with Bioconductor
- Software best practices (testing, documentation, CI/CD)
- Web development for interfaces

Collaborators/Advisors

Statistical Advisor:

- Expert in high-dimensional statistics
- Preferably with RUV method experience
- Could be original RUV authors (Speed, Gagnon-Bartsch)

Clinical/Cancer Biologist:

- Validates biological relevance of results
- Provides use cases and test datasets
- Interprets findings in clinical context

User Experience (UX) Specialist:

- Designs intuitive interfaces
- Conducts user studies
- Ensures accessibility for non-experts

Ideal Collaborative Structure

- Academic lab (method development, validation)

- Partner with RUV authors (Molania, Speed at WEHI)
- Industry partner (real-world validation, sustainability)
- Computing center (GPU resources, cloud credits)

Preliminary Data and Feasibility

Evidence Supporting Feasibility

1. **RUV-III Success:** Molania et al. (2023) demonstrates that statistical framework works
    - Provides gold standard for comparison
    - Extensive benchmarking already done
    - Clear metrics for success defined
2. **AI in Adjacent Areas:** Successful ML applications in related tasks
    - Gene selection: RF and VAEs work well
    - Batch correction: Deep learning approaches emerging
    - Single-cell: scVI, scANVI show promise
3. **Data Availability:** Large public datasets for training
    - TCGA: 11,000 samples, extensively characterized
    - GTEx: 17,000 samples, normal tissue baseline
    - GEO: Thousands of additional studies
    - Ground truth from technical replicates
4. **Computational Resources:** Feasible hardware requirements
    - Models trainable on single GPU workstation
    - Inference fast even on CPU
    - Cloud options for large-scale training
5. **Software Ecosystem:** Strong foundation to build on
    - Bioconductor infrastructure
    - PyTorch/TensorFlow mature
    - RUVSeq package as starting point
    - Active community support

Preliminary Results (If Available)

- Simple ML model for NCG selection shows promise (if pilot data exists)
- Clustering algorithms identify reasonable biological groups
- Transfer learning improves small sample performance in toy examples

---

Where to Seek Help and Resources

Funding Opportunities

United States:

NIH (National Institutes of Health):

- R01 Research Grant: $250K-$500K/year for 4-5 years
    - Relevant Institutes: NCI (cancer), NHGRI (genomics), NIGMS (general methods)
    - Program: Computational Methods and Software Development
- R21 Exploratory Grant: $200K total for 2 years (pilot funding)
    - Good for initial proof-of-concept
- R03 Small Grant: $100K total for 2 years

- SBIR/STTR: If commercialization planned (industry partnership)

NSF (National Science Foundation):

- Division of Mathematical Sciences (DMS): Statistics methods
- Division of Computing and Communication Foundations (CCF): Machine learning
- Division of Biological Infrastructure (DBI): Bioinformatics tools
- Typical funding: $150K-$500K for 3 years

Private Foundations:

- Chan Zuckerberg Initiative: Computational biology, open science
- Alfred P. Sloan Foundation: Data science, computational methods
- Gordon and Betty Moore Foundation: Data-driven discovery
- Simons Foundation: Basic science, computational biology
- Typical funding: $100K-$1M

Industry Partnerships:

- Pharmaceutical companies: Need robust multi-site normalization
  - Roche, Novartis, Pfizer, Merck
- Biotech companies: Developing RNA-seq diagnostics
  - Illumina, 10x Genomics, Pacific Biosciences
- Tech companies: AI/ML in healthcare
  - Google Health, Microsoft Healthcare, AWS HealthAI

International:

- European Research Council (ERC): Starting/Consolidator grants (€1-2M)
- Wellcome Trust (UK): Methods development grants
- Australian Research Council (ARC): Discovery Projects
- Canadian Institutes of Health Research (CIHR)

Collaborative Opportunities

Connect with RUV Authors:

- Ramyar Molania: molania.r@wehi.edu.au (Dana-Farber Cancer Institute)
  - Lead author, likely very knowledgeable about limitations and opportunities
  - May be interested in AI collaboration
- Terence P. Speed: speed@wehi.edu.au (Walter & Eliza Hall Institute)
  - Senior author, statistics expert, may advise or collaborate
- Johann Gagnon-Bartsch: jgagnon@umich.edu (University of Michigan)
  - Original RUV-III developer, statistical foundations

Academic Institutions Strong in Computational Biology + AI:

- Broad Institute (Harvard/MIT): Computational biology powerhouse
- Stanford University: Biomedical Data Science Department
- UCSF: Institute for Computational Health Sciences
- Carnegie Mellon University: Computational Biology Department
- UC San Diego: Bioinformatics and Systems Biology
- Cold Spring Harbor Laboratory: Cancer genomics and bioinformatics

Existing Consortia/Initiatives:

- NCI ITCR (Informatics Technology for Cancer Research): Fund cancer informatics tools

- TOPMed (Trans-Omics for Precision Medicine): Multi-omics integration
- GA4GH (Global Alliance for Genomics and Health): Standards and tools
- Human Cell Atlas: Single-cell data standardization

Technical Resources

Computing Resources:

- NSF XSEDE/ACCESS: Free supercomputing time for academic research
- NIH Biowulf: HPC cluster for NIH-funded researchers
- Google Cloud for Research: Cloud credits for academic projects
- AWS Educate/Research: Cloud credits and training
- Microsoft Azure for Research: Cloud grants for academic projects

Training and Learning:

Deep Learning for Genomics Courses:

- Stanford CS273B (Deep Learning in Genomics)
- MIT 6.047/6.878 (Computational Biology)
- Coursera: Deep Learning Specialization + Genomic Data Science

Workshops:

- Cold Spring Harbor Laboratory: Computational Genomics courses
- Marine Biological Laboratory: Bioinformatics workshops

Online Communities:

- Bioconductor Support Forum
- Biostars Q&A
- r/bioinformatics Reddit
- Computational Biology Slack/Discord servers

Software and Tools:

- Deep Learning Frameworks: PyTorch, TensorFlow/Keras
- Bioinformatics: Bioconductor, Scanpy (Python)
- Experiment Tracking: MLflow, Weights & Biases
- Reproducibility: Docker, Conda, Snakemake

Community Engagement

Conferences to Present/Network:

- ISMB (Intelligent Systems for Molecular Biology): Premier computational biology
- RECOMB (Research in Computational Molecular Biology): Methods focus
- ASHG (American Society of Human Genetics): Clinical genomics
- NeurIPS/ICML/ICLR: Machine learning (Bio-ML workshops)
- Bioinformatics Open Source Conference (BOSC): Software development

Journals for Publication:

- Methods: Nature Biotechnology, Nature Methods, Genome Biology
- Bioinformatics: Bioinformatics, BMC Bioinformatics, Nucleic Acids Research
- Machine Learning: NeurIPS, ICML, ICLR (AI/ML venues)

- Hybrid: PLOS Computational Biology, Cell Systems

Social Media and Outreach:

- Twitter/X: #bioinformatics, #compbio, #machinelearning hashtags
- LinkedIn: Network with industry partners
- YouTube: Tutorial videos gain visibility
- Blog Posts: Medium, personal website for methods explanations

---

Risk-Benefit Analysis

Risk Assessment

Scientific Risks:

- Low Risk: Basic feasibility - AI works in adjacent areas, RUV-III framework solid
- Medium Risk: Transfer learning effectiveness - may not generalize as well as hoped
- Medium Risk: Interpretability acceptance - biologists may resist black boxes
- Low Risk: Computational feasibility - hardware requirements reasonable

Technical Risks:

- Low Risk: Software development - established tools and frameworks
- Medium Risk: Scalability - may need optimization for very large datasets
- Low Risk: Integration - R/Python ecosystems well-developed

Adoption Risks:

- Medium Risk: Changing established workflows - researchers may stick with familiar tools
- Low Risk: Documentation quality - can control through effort
- Medium Risk: Maintenance burden - requires long-term commitment

Overall Risk Level: LOW to MEDIUM

- Most components have been proven in adjacent applications
- Clear validation path with TCGA benchmark
- Strong foundation from Molania et al. work

Benefit Assessment

Scientific Benefits:

- High: Automated, reproducible normalization reduces researcher burden
- High: Better normalization → more accurate downstream analyses
- High: Enables small labs to achieve TCGA-level quality
- Medium: New insights from detecting unknown batch effects

Clinical Benefits:

- High: Better biomarkers for patient stratification
- Medium: Enables multi-center clinical trials with transcriptomics
- High: Improves reproducibility of precision medicine

Economic Benefits:

- Medium: Reduces time spent on normalization (researcher productivity)
- Low-Medium: Potential for commercialization/licensing
- High: Industry partners would pay for robust multi-site normalization

Community Benefits:

- High: Open-source tool benefits entire field
- Medium: Training materials help next generation
- High: Sets precedent for AI in established statistical frameworks

Overall Benefit Level: HIGH

- Clear unmet need (manual RUV-III is tedious)
- Large potential user base (anyone doing RNA-seq)
- Multiple stakeholder benefits

Risk-Benefit Conclusion

Favorable Risk-Benefit Ratio

- Risks: Manageable, mostly in adoption/generalization
- Benefits: Substantial, wide-reaching impact
- **Recommendation:** Project is well-justified and should proceed

---

Next Steps and Action Items

Immediate Actions (Next 1-3 Months)

For Researchers Interested in Pursuing This:

1. **Contact RUV Authors**
   - Email Ramyar Molania expressing interest in AI enhancement
   - Request access to RUVprps package development
   - Discuss potential collaboration or mentorship
2. **Assemble Initial Team**
   - Identify ML expert collaborator
   - Recruit bioinformatics graduate student or postdoc
   - Secure computational resources (GPU access)
3. **Preliminary Data Collection**
   - Download TCGA READ, COAD, BRCA datasets
   - Implement baseline RUV-III using RUVSeq package
   - Establish benchmark metrics (reproduce Molania results)
4. **Proof-of-Concept Prototype**
   - Develop simple ML model for NCG selection
   - Test on TCGA READ data
   - Compare to expert-selected NCGs
   - Document results for grant preliminary data
5. **Grant Preparation**
   - Identify target funding mechanism (R21, R01, NSF)
   - Draft specific aims
   - Prepare preliminary figures
   - Get feedback from mentors/collaborators

Medium-Term Actions (3-12 Months)

6. **Expand Prototypes**
   - Develop clustering algorithm for PRPS
   - Test K optimization approaches
   - Validate on multiple TCGA cancers
7. **Build Collaborations**
   - Visit WEHI or Speed lab if possible
   - Present at lab meetings to get feedback
   - Recruit clinical collaborators for validation
8. **Submit Grants**
   - Submit R21 or NSF proposal
   - Apply for cloud computing credits
   - Seek foundation pilot funding
9. **Begin Publications**
   - Write methods paper on NCG selection
   - Submit to bioRxiv as preprint
   - Present at regional conferences
10. **Software Development**
    - Start R package structure
    - Implement version control (GitHub)
    - Begin documentation

Long-Term Actions (1-4 Years)

11. **Full Implementation**
    - Complete all 6 objectives
    - Extensive validation
    - User studies
12. **Dissemination**
    - Publish in high-impact venues
    - Release production software
    - Host workshops
13. **Sustainability**
    - Build user community
    - Secure maintenance funding
    - Train contributors

Conclusion

Summary of Opportunity

The integration of AI/ML with RUV-III normalization represents a high-impact, feasible research opportunity with:

- **Clear Need:** Manual RUV-III is tedious, requires expertise
- **Open Field:** No existing AI-enhanced RUV methods
- **Strong Foundation:** RUV-III proven effective, extensive benchmarks available
- **Technical Feasibility:** Adjacent AI applications successful, computational resources available
- **Broad Impact:** Benefits cancer research, clinical transcriptomics, precision medicine
- **Multiple Stakeholders:** Academic, clinical, industry interest

Key Innovations

This proposal would represent the first comprehensive AI-enhanced normalization framework that:

1. Automates expert decision-making while maintaining interpretability
2. Detects unknown batch effects using deep learning
3. Enables transfer learning across studies and cancer types
4. Provides end-to-end automated pipeline
5. Integrates multiple AI techniques (supervised, unsupervised, RL, transfer learning)

Expected Impact

Within Bioinformatics:

- Sets precedent for AI enhancement of statistical methods
- Demonstrates successful ML/statistics integration
- Provides template for other normalization methods

Within Cancer Research:

- Improves TCGA data re-analysis
- Enables more robust multi-site clinical trials
- Accelerates biomarker discovery

Within Precision Medicine:

- Better patient stratification
- More reliable prognostic signatures
- Cross-platform biomarker translation

Final Recommendation

This project should be pursued. The combination of:

- Clear unmet need
- Technical feasibility
- Available resources
- Strong preliminary work (Molania et al.)
- Favorable risk-benefit ratio
- Multiple funding pathways

...makes this an excellent opportunity for researchers interested in AI/bioinformatics methods development.

Success would be a significant contribution to:

- Computational biology methodology
- Cancer genomics practice
- Clinical transcriptomics
- Open-source bioinformatics tools

Document Status: Complete - Part 2 of 2
Previous Document: Part 1 - Current State of AI/ML in RUV Normalization