

1 **ResistanceGA: An R package for the optimization of resistance surfaces using genetic**  
2 **algorithms**

3

4 **Running Title:** Landscape resistance optimization

5

6 **Author:** William E. Peterman

7

8 **Address:** Illinois Natural History Survey, Prairie Research Institute, University of Illinois, 1816

9 S. Oak Street, Champaign, IL 61820 USA

10

11 **Email:** [Bill.Peterman@gmail.com](mailto:Bill.Peterman@gmail.com); **Phone:** 217.244.8191

12

13 **Word count:** 3,000

14

## 15 **Summary**

- 16 1. Understanding how landscape features affect functional connectivity among populations  
17 is a cornerstone of landscape genetic analyses. However, parameterization of resistance  
18 surfaces that best describe connectivity is largely a subjective process that explores a  
19 limited parameter space.
- 20 2. `ResistanceGA` is a new R package that utilizes a genetic algorithm to optimize  
21 resistance surfaces based on pairwise genetic distances and either `CIRCUITSCAPE`  
22 resistance distances or cost distances calculated along least cost paths. Functions in this  
23 package allow for the optimization of both categorical and continuous resistance surfaces,  
24 as well as the simultaneous optimization of multiple resistance surfaces.
- 25 3. There is considerable controversy concerning the use of Mantel tests to accurately relate  
26 pairwise genetic distances with resistance distances. Optimization in `ResistanceGA`  
27 uses linear mixed effects models with the maximum likelihood population effects  
28 parameterization to determine AICc, which is the fitness function for the genetic  
29 algorithm.
- 30 4. `ResistanceGA` fills a void in the landscape genetic toolbox, allowing for unbiased  
31 optimization of resistance surfaces and for the simultaneous optimization of multiple  
32 resistance surfaces to create a novel composite resistance surface.

33

34 **Key-words:** `CIRCUITSCAPE`, circuit theory, cost distance, gene flow, genetic algorithm,  
35 landscape genetics, least cost path, resistance distance, resistance optimization, resistance surface

## 36 **Introduction**

37 First coined in 2003, landscape genetics has experienced rapid growth in both the number of  
38 studies and range of analytical methods utilized (Manel *et al.* 2003; Storfer *et al.* 2010). This  
39 integrative field draws on landscape ecology, spatial statistics, and population genetics to address  
40 a wide range of questions. One of the primary goals of landscape genetic studies has been to  
41 understand how landscape features affect spatial genetic structure (Manel *et al.* 2003; Storfer *et*  
42 *al.* 2007). Rather than simply assessing isolation-by-distance, many landscape genetic studies  
43 quantify the effective distance (i.e. resistance distance) between spatially distinct sample  
44 locations as a function of the landscape matrix (Spear *et al.* 2005; McRae 2006). In the absence  
45 of direct observation of movement or dispersal across the landscape, resistance distances are  
46 often interpreted as functional connectivity (e.g., Cushman *et al.* 2006). Critically, to infer  
47 functional connectivity and resistance distance, an appropriate resistance surface must be  
48 parameterized. As defined by Spear *et al.* (2010), a resistance surface is a spatial layer that  
49 assigns a value to each landscape or environmental feature, with values representing the extent to  
50 which that feature impedes or facilitates connectivity for an organism.

51 Resistance values of surfaces have been determined using a variety of methods,  
52 including: habitat suitability models (e.g., Wang *et al.* 2008), telemetry (e.g., Driezen *et al.*  
53 2007), and most commonly expert opinion (Murray *et al.* 2009). Less often, parameterization is  
54 informed by empirical movement studies (e.g., Stevens *et al.* 2006) or spatial prediction of  
55 ecological processes (Peterman *et al.* 2014). All of these are acceptable approaches, but each  
56 comes with its own caveats. Of particular concern is the fact that expert opinion often fails to  
57 accurately describe the biological or ecological process(es) being modeled (Shirk *et al.* 2010;  
58 Charney 2012). Even when biological or ecological processes are known and explicitly modeled,

59 there is no guarantee that these processes will relate meaningfully to the movement of genes  
60 across the landscape (Peterman *et al.* 2014).

61       Ultimately, the assignment and evaluation of resistance values generally covers only a  
62 limited parameter space and remains a trial and error process to determine the best resistance  
63 parameters. Graves, Beier and Royle (2013) attempted to alleviate the subjectivity of resistance  
64 surface parameterization by using a search algorithm to maximize Mantel  $r$  correlation between  
65 inter-individual genetic distance and least cost path distance. While their optimization procedures  
66 recovered the maximum Mantel  $r$  when it existed, they found that resistance estimates were often  
67 imprecise and much smaller than simulated resistance values. They also found that response  
68 surfaces were quite flat, making identification of a global optimum difficult. In contrast,  
69 Peterman *et al.* (2014) found well-defined global optima when using Ricker and monomolecular  
70 data transformations in combination with optimization algorithms. However, the optimization  
71 procedure utilized by Peterman *et al.* (2014) is limited to continuous resistance surfaces (e.g.,  
72 temperature, moisture, percent canopy cover) and requires an inefficient search of all possible  
73 data transformations.

74       There are numerous challenges to optimizing resistance surfaces based on pairwise  
75 (genetic) distance data. Among these challenges is the fact that resistance surfaces can have a  
76 high dimensionality, as in land use, land cover surfaces. Another issue is that there currently is  
77 no closed-form expression to determine the landscape resistance values that describe pairwise  
78 distances, potentially making optimization intractable with gradient-based algorithms. Finally,  
79 landscape features and environmental gradients do not exist in isolation. Therefore, an ideal  
80 solution to resistance surface optimization will be to simultaneously optimize multiple resistance  
81 surfaces to create a composite resistance surface. The `ResistanceGA` package for the R

82 programming environment (R Core Team 2014) has been developed to address these issues, and  
83 to fill a void in the landscape genetic toolbox. While the initial impetus for this package was  
84 landscape genetic analyses, but any pairwise measures across the landscape (e.g., movement  
85 rates) could be utilized to optimize resistance surfaces.

86

## 87 **Description**

### 88 **Genetic algorithms**

89 Genetic algorithms (GAs) represent a powerful and flexible stochastic optimization framework  
90 for finding solutions to both discrete and continuous optimization problems (Holland 1975).  
91 Inspired by biological principles, genetic algorithms create a population of individuals  
92 (offspring) with traits (parameters to be optimized) encoded on “chromosomes”. The genotypes  
93 (parameter combinations) of each individual solve the fitness function, and the fittest individuals  
94 from each generation survive to reproduce (Sivanandam & Deepa 2007). The GA evolution  
95 process is facilitated by exploration and exploitation (Scrucca 2013). Exploration of parameter  
96 space occurs through random generation of new parameter values resulting from mutation, as  
97 well as exchange of genetic information through crossover. Exploitation ultimately reduces  
98 diversity in the population by selecting the fittest individuals each generation. The population  
99 continues to evolve until a sufficient number of generations have passed without an improvement  
100 in fitness (Scrucca 2013).

### 101 **Resistance optimization**

102 `ResistanceGA` utilizes the general-purpose genetic algorithm from the GA R package (`ga`  
103 function; Scrucca 2013). Briefly, the optimization proceeds as follows:

- 104 1) The original raster surface is imported into R. If the surface is continuous, it is rescaled to  
105 range of 0–10, preserving the relative spacing between all levels.
- 106 2) The evolution process starts by generating a random initial population of size  $n$ . If a  
107 continuous surface, the selected parameters determine (a) which of eight transformations  
108 will be applied (Fig. 1); (b) the shape of the transformation; (c) the maximum value of the  
109 transformation. If a categorical surface, each level of the resistance surface is reclassified  
110 to the values of the parameters.
- 111 3) Using the spatial locations where genetic samples have been collected, either  
112 CIRCUITSCAPE (McRae *et al.* 2008; McRae & Shah 2009) is called from R to calculate  
113 pairwise resistance distances across the landscape created in step 2 above, or cost  
114 distances are calculated from least cost paths using the R package `gdistance` (van  
115 Etten 2014).
- 116 4) A linear mixed effects model with a maximum likelihood population effects  
117 parameterization (MLPE) is fit to the data. Pairwise genetic distance is the response and  
118 the scaled and centered pairwise resistance distance is the predictor. The MLPE mixed  
119 effects parameterization is used to account for the non-independence among the pairwise  
120 data (Clarke, Rothery & Raybould 2002).
- 121 5) The Akaike information criterion (AIC) is obtained from the fitted MLPE model. AICc is  
122 then calculated by penalizing the model for the number of parameters used during  
123 transformation/reclassification. AICc is the fitness measure that the genetic algorithm  
124 seeks to maximize. Because the `ga` function works through maximization, the sign of  
125 AICc is reversed.

- 126       6) Steps 2–5 are repeated until the specified number of  $n$  individuals have been created. The  
127       genetic algorithm then conducts selection on the population, and the individuals with the  
128       best AICc are carried over to the next generation to form the reproducing population. A  
129       new population is then formed through mutation and crossover.
- 130       7) Steps 2–6 are repeated until the specified number of generations have passed without  
131       improvement to the AICc.

132

### 133       **Continuous surfaces**

134       There are eight transformations that can be applied to continuous surfaces (Fig. 1). Each  
135       transformation relies on either the Ricker or monomolecular functions (Bolker 2008), as well as  
136       rescaling functions to keep values in positive parameter space . The shape and magnitude of each  
137       transformation are controlled by two parameters, which are adjusted during optimization  
138       (Peterman *et al.* 2014). During optimization, the genetic algorithm searches different  
139       combinations of transformations, scale parameters, and shape parameters. Linear transformations  
140       are not explicitly included, but all of the monomolecular functions become linear as the shape  
141       parameter increases in value. In this way, linear responses can effectively be modeled without  
142       increasing the number of transformations for the genetic algorithm to test.

### 143       **Categorical surfaces**

144       Categorical or feature surfaces, such as land cover or roads, can also be optimized using  
145       ResistanceGA. A surface is considered categorical in ResistanceGA if it contains 15 or fewer  
146       levels. To make this process tractable, it is necessary to hold the value of one level constant  
147       throughout optimization. Because pairwise resistance values are relative, failing to hold one level

148 constant can result in multiple equivalent solutions, and the algorithm may fail to reach an  
149 optimal solution (e.g., relative resistance values of 1, 5, and 10 are equivalent to 2, 10, and 20).

### 150 **Combining resistance surfaces**

151 Resistance surfaces are simultaneously optimized by sequentially modifying each surface to be  
152 optimized. Continuous and categorical surfaces are each modified as described above, then all  
153 modified surfaces are summed together to create a single, composite resistance surface, which is  
154 then used to calculate pairwise resistance distances or cost distances.

155

### 156 **Overview of ResistanceGA functions**

157 The optimization functions passed to `ga` rely heavily on the R package `raster` (Hijmans 2014)  
158 to import, export and manipulate spatial raster (*.asc*) files, as well as `lme4` (Bates *et al.* 2014) to  
159 fit mixed effects models. To visualize continuous surface transformations, `ResistanceGA` uses  
160 the graphing features of `ggplot2` (Wickham 2009). The functions available in `ResistanceGA`  
161 are summarized in Table 1.

162

### 163 **Implementation**

164 Resistance surfaces can be optimized using cost distances (least cost path) or using circuit-based  
165 resistance distances. Cost distances are calculated using the R package `gdistance` (van Etten  
166 2014), while electrical current resistances are calculated using the free, open-source software  
167 `CIRCUITSCAPE` (v4.0 or greater; McRae & Shah 2009). If using `CIRCUITSCAPE`, input data  
168 formats must conform to those of `CIRCUITSCAPE` (McRae & Shah 2009). Currently,  
169 `CIRCUITSCAPE` can only be executed from R through `ResistanceGA` on computers using  
170 Windows operating systems. Prior to running any of the optimization functions (`MS_optim`,



171 `SS_optim`, `Resistance.Opt_multi` or `Resistance.Opt_single`) the  
172 `CS.prep/gdist.prep`, and `GA.prep` preparation functions must be run. These functions create  
173 and format the inputs and data objects necessary to run `CIRCUITSCAPE/gdistance`,  
174 parameterize the genetic algorithm, and fit MLPE mixed effects models. These functions have  
175 been developed to require a minimum input from the user. For instance, at minimum, all that  
176 needs to be specified to run `GA.prep` is the directory where the `.asc` files to be optimized reside.  
177 However, all available arguments of the `ga` function can be set by the user to modify the genetic  
178 algorithm. The arguments and default settings of the preparation functions are described in Table  
179 2.

180       There is no established framework for optimization of resistance surfaces, but an  
181 envisioned work flow is detailed in Figure 2. Genetic algorithms are stochastic optimization  
182 procedures; therefore it is highly advised to run all optimizations at least twice to confirm  
183 convergence and parameter estimates. Also, because boundaries are placed on the parameter  
184 space searched, if the optimized resistance values are at or near the limits set, the optimization  
185 should be rerun after expanding the search space. It is important to note that simultaneous  
186 optimization of multiple surfaces often results in parameter values that differ from truth.  
187 However, the relative resistance values of the resultant resistance surface are highly or perfectly  
188 correlated with truth (see worked example), and the resistance surfaces are identical once they  
189 are rescaled to a minimum resistance of one. Users of `ResistanceGA` should be aware of this  
190 fact, and interpret results accordingly.

191       Genetic algorithms are effective at finding an optimal solution, but they can be  
192 computationally intensive. To ensure that parameter space is adequately searched, the population  
193 of individuals produced each generation must be of sufficient size. In `ResistanceGA` the default

194 setting is to produce a population that is 15 times the number of parameters being optimized, up  
195 to a maximum population size of 100 individuals. For example, when a continuous surface is  
196 being optimized, 45 individuals (3 parameters x 15) will be produced each generation, and a  
197 typical optimization takes 50–300 generations. This results in the creation of 2 250–13 500  
198 resistance surfaces and CIRCUIITSCAPE/*gdistance* runs. Therefore, the greatest impediment  
199 to optimizing resistance surfaces is time. Both the spatial extent and the number of sample  
200 locations in the analysis affect the time necessary to calculate pairwise resistance or cost  
201 distances. On a computer with an Intel i7 3.4 GHz processor, 25 sample locations distributed  
202 across a 250 x 250 cell resistance surface takes ~5 seconds to complete an iteration with  
203 CIRCUIITSCAPE. The same surface with 100 sample locations takes ~14 seconds. Finally, 25  
204 sample locations on and a 1 000 x 1 000 cell surface can take up to 90 seconds to complete.  
205 Because CIRCUIITSCAPE calculates resistance over all possible pathways, it is more  
206 computationally intensive than calculating least cost paths. On average, optimization with  
207 *gdistance* is about three times faster than calculating resistance distances with  
208 CIRCUIITSCAPE. To further reduce the optimization time when using least cost paths, the  
209 optimization can be run in parallel by setting `parallel = TRUE` in `GA.prep`. Additional  
210 strategies to reduce the runtime of CIRCUIITSCAPE/*gdistance* include modifying the  
211 connection scheme (`Neighbor.Connect/directions`) from the default setting of 8 to 4, and  
212 reducing the resolution of the resistance surfaces (i.e. increase cell size). The latter will reduce  
213 the total number of cells on the surface, and generally has limited effect on the quantification of  
214 relative resistances across the landscape (McRae *et al.* 2008).

215

216 **Worked examples**

217 The examples below use simulated data provided with `ResistanceGA`. Code to generate the  
218 example resistance surfaces and spatial point data can be found in Appendix S1. More extensive  
219 examples and further descriptions of the functions included in `ResistanceGA` can be found in  
220 the vignette included with the package.

221

### 222 **Example 1: Single surface optimization**

223 This example optimizes a single resistance surface using `CIRCUITSCAPE`.

```
224 # Get data
```

```
225 # Install 'devtools' package, if needed
```

```
226 if(!("devtools" %in% list.files(.libPaths())) {
```

```
227     install.packages("devtools")
```

```
228 }
```

229

```
230 library(devtools)
```

```
231 install_github("wpeterman/ResistanceGA") # Download package
```

```
232 library(ResistanceGA) # Load package
```

```
233 rm(list = ls())
```

234

```
235 # Create directory for reading/writing CIRCUITSCAPE files and results
```

```
236 if("ResistanceGA_Examples"%in%dir("C:/")==FALSE)
```

```
237     dir.create(file.path("C:/", "ResistanceGA_Examples"))
```

238

```
239 # Create a subdirectory for the first example
```

```
240 dir.create(file.path("C:/ ResistanceGA_Examples/", "SingleSurface"))
```

241

```
242 # Directory to write .asc files and results
```

```
243 write.dir <- "C:/ ResistanceGA_Examples/SingleSurface/"
```

244

```
245 # Load example data, write continuous surface as `.asc` file
```

```
246 data(resistance_surfaces)
```

```
247 continuous <- resistance_surfaces[[2]]
248 writeRaster(continuous,filename=paste0(write.dir,"cont.asc"),overwrite=TRUE)
249
250 # Load sample location data
251 data(samples)
252 write.table(samples,
253             file=paste0(write.dir,"samples.txt"),
254             sep="\t",col.names=F,row.names=F)
255
256 # Create a spatial points object for plotting
257 sample.locales <- SpatialPoints(samples[,c(2,3)])
258
259 # Run preparation functions
260 GA.inputs <- GA.prep(ASCII.dir=write.dir,
261                    max.cat=500,
262                    max.cont=500,
263                    seed=555,
264                    quiet=TRUE)
265
266 CS.inputs <- CS.prep(n.POPS=length(sample.locales),
267                    CS_Point.File=paste0(write.dir,"samples.txt"),
268                    CS.program="'C:/Program Files/Circuitscape/cs_run.exe'')
269
270 # Transform the continuous surface
271 r.tran <- Resistance.tran(transformation="Monomolecular",
272                          shape=2,
273                          max=275,
274                          r=continuous)
275
276 # Visualize the transformation (Fig. 3)
277 plot.t <- Plot.trans(PARM=c(2,275),
278                    Resistance=continuous,
```

```
279         transformation="Monomolecular")
280
281 # Calculate pairwise resistance values to use as the response
282 CS.response <- Run_CS(CS.inputs=CS.inputs,
283                      GA.inputs=GA.inputs,
284                      r=r.tran)
285
286 # Rerun `CS.prep` to include response
287 CS.inputs <- CS.prep(n.POPS=length(sample.locales),
288                    response=CS.response,
289                    CS_Point.File=paste0(write.dir,"samples.txt"),
290                    CS.program='"C:/Program Files/Circuitscape/cs_run.exe"')
291
292 # Run optimization
293 SS_RESULTS <- SS_optim(CS.inputs=CS.inputs,
294                      GA.inputs=GA.inputs)
295
296 # Compare results
297 SS_table <- data.frame(c("Monomolecular", 2.0, 275),
298                      t(SS_RESULTS$ContinuousResults[c(3:5)]))
299 colnames(SS_table) <- c("Truth", "Optimized")
300 SS_table
301           Truth      Optimized
302 Equation Monomolecular Monomolecular
303 shape           2      1.999999
304 max            275      274.9982
305
```

306 The exact transformation parameters were recovered in 147 iterations (Fig 3).

### 307 **Example 2: Multisurface optimization**

308 This example simultaneously optimizes three resistance surface using least cost paths.

```
309 # Get data
```

```
310 data(resistance_surfaces)
311 data(samples)
312
313 # Create a spatial points object
314 sample.locales <- spatialPoints(samples[,c(2,3)])
315
316 # Create a subdirectory for the second example
317 dir.create(file.path("C:/ResistanceGA/", "MultipleSurfaces"))
318
319 # Run `GA.prep`
320 GA.inputs <- GA.prep(ASCII.dir=resistance_surfaces,
321                      Results.dir="C:/ResistanceGA/MultipleSurfaces/",
322                      max.cat=500,
323                      max.cont=500,
324                      seed = 321,
325                      parallel = 4) # Run on in parallel on 4 cores
326
327 # Run `CS.prep` functions
328 gdist.inputs<-gdist.prep(n.POPS=length(sample.locales),
329                          samples=sample.locales)
330
331 # Set parameters for transforming and combining resistance surfaces
332 PARM <- c(1,250,75,6,3.5,150,1,350)
333
334 # PARM<- c(1, # First feature of categorical
335 #          250, # Second feature of categorical
336 #          75, # Third feature of categorical
337 #          6, # Transformation equation for continuous surface
338 #          3.5, # Shape parameter
339 #          150, # Scale parameter
340 #          1, # First feature of feature surface
341 #          350) # Second feature of feature surface
```

```
342
343 # Combine resistance surfaces
344 Resist <- Combine_Surfaces(PARM=PARM,
345                             gdist.inputs=gdist.inputs,
346                             GA.inputs=GA.inputs,
347                             out=NULL,
348                             rescale=TRUE)
349
350 # Create the response surface
351 gd.response <- Run_gdistance(gdist.inputs=gdist.inputs,
352                              GA.inputs=GA.inputs,
353                              r=Resist)
354
355 # Re-run `gdist.prep` to include the response
356 gdist.inputs<-gdist.prep(n.POPS=length(sample.locales),
357                          response=lower(as.matrix(gd.response)),
358                          samples=sample.locales)
359
360 # Run `MS_optim`
361 Multi.Surface_optim.gd <- MS_optim(gdist.inputs=gdist.inputs,
362                                    GA.inputs=GA.inputs)
363
364 # View optimized parameter values with the true simulation values
365 Summary.table <- data.frame(PARM,round(t(Multi.Surface_optim.gd@solution),2))
366 colnames(Summary.table)<-c("Truth", "Optimized")
367 row.names(Summary.table)<-c("Category1", "Category2", "Category3",
368 "Transformation", "Shape", "Max", "Feature1", "Feature2")
369 Summary.table
370           Truth Optimized
371 Category1      1.0      1.00
372 Category2    250.0    268.27
373 Category3     75.0     80.42
```

```
374 Transformation    6.0      6.51
375 Shape              3.5      3.50
376 Max                150.0    161.11
377 Feature1           1.0      1.00
378 Feature2           350.0    375.56
```

379

```
380 # Compare the true and optimized resistance surfaces
381 optim.resist <- Combine_surfaces(PARM=Multi.Surface_optim.gd@solution,
382                                 gdist.inputs = gdist.inputs,
383                                 GA.inputs = GA.inputs)
384
385 resist.stack <- stack(Resist,optim.resist)
386 names(resist.stack) <- c("Truth", "Optimized")
387 pairs(resist.stack) # Correlation plot (Fig. 4)
388 plot(resist.stack) # Resistance maps (Fig. 4)
```

389

390 This example took 210 iterations to optimize. As described above, the optimization has  
391 recovered different parameter values than those that were simulated, but the relative values of the  
392 simulated and optimized surfaces are identical (Fig. 4).

393

### 394 **Obtaining ResistanceGA**

395 ResistanceGA is hosted on GitHub, and can be downloaded using the `install.github`  
396 function from the `devtools` package:

```
397 install.github("wpeterman/ResistanceGA")
```

398 Following download, the package can be loaded:

```
399 library(ResistanceGA)
```

400 View the vignette with more demonstrations of ResistanceGA functions:

```
401 vignette('ResistanceGA')
```



402 Alternatively, the HTML vignette can be viewed here: <http://goo.gl/n7VOZx>

### 403 **Acknowledgements**

404 I thank G. Connette for many discussions concerning the development and implementation of  
405 functions in `ResistanceGA`, and for comments on an early draft of this manuscript.

406

### 407 **Data accessibility**

408 Example data are provided with `ResistanceGA` or can be made using the code provided in  
409 Appendix S1.

410

### 411 **Supporting information**

412 Appendix S1. R code to generate the example resistance surfaces provided with `ResistanceGA`

413 **References**

- 414 Bates, D.M., Maechler, M., Bolker, B.M. & Walker, S. (2014) lme4: Linear mixed-effects  
415 models using Eigen and S4. R package version 1.1-6. [http://CRAN.R-](http://CRAN.R-project.org/package=lme4)  
416 [project.org/package=lme4](http://CRAN.R-project.org/package=lme4).
- 417 Bolker, B.M. (2008) *Ecological Models and Data in R*. Princeton University Press, Princeton,  
418 NJ.
- 419 Charney, N.D. (2012) Evaluating expert opinion and spatial scale in an amphibian model.  
420 *Ecological Modelling*, **242**, 37–45.
- 421 Clarke, R.T., Rothery, P. & Raybould, A.F. (2002) Confidence limits for regression relationships  
422 between distance matrices: Estimating gene flow with distance. *Journal of Agricultural,*  
423 *Biological, and Environmental Statistics*, **7**, 361–372.
- 424 Cushman, S.A., McKelvey, K.S., Hayden, J. & Schwartz, M.K. (2006) Gene flow in complex  
425 landscapes: Testing multiple hypotheses with causal modeling. *American Naturalist*, **168**,  
426 486–499.
- 427 Driegen, K., Adriaensen, F., Rondinini, C., Doncaster, C.P. & Matthysen, E. (2007) Evaluating  
428 least-cost model predictions with empirical dispersal data: A case-study using  
429 radiotracking data of hedgehogs (*Erinaceus europaeus*). *Ecological Modelling*, **209**, 314–  
430 322.
- 431 Graves, T.A., Beier, P. & Royle, J.A. (2013) Current approaches using genetic distances produce  
432 poor estimates of landscape resistance to interindividual dispersal. *Molecular Ecology*,  
433 **22**, 3888–3903.
- 434 Hijmans, R.J. (2014) raster: Geographic data analysis and modeling. R package version 2.2-31.  
435 <http://CRAN.R-project.org/package=raster>.

- 436 Holland, J.H. (1975) *Adaptation In Natural And Artificial Systems: An Introductory Analysis*  
437 *With Applications To Biology, Control, And Artificial Intelligence*. University of  
438 Michigan Press.
- 439 Manel, S., Schwartz, M.K., Luikart, G. & Taberlet, P. (2003) Landscape genetics: combining  
440 landscape ecology and population genetics. *Trends in Ecology and Evolution*, **18**, 189–  
441 197.
- 442 McRae, B.H. (2006) Isolation by resistance. *Evolution*, **60**, 1551–1561.
- 443 McRae, B.H., Dickson, B.G., Keitt, T.H. & Shah, V.B. (2008) Using circuit theory to model  
444 connectivity in ecology, evolution, and conservation. *Ecology*, **89**, 2712–2724.
- 445 McRae, B.H. & Shah, V.B. (2009) Circuitscape user's guide. ONLINE. The University of  
446 California, Santa Barbara. Available at: <http://www.circuitscape.org>.
- 447 Murray, J.V., Goldizen, A.W., O'Leary, R.A., McAlpine, C.A., Possingham, H.P. & Choy, S.L.  
448 (2009) How useful is expert opinion for predicting the distribution of a species within and  
449 beyond the region of expertise? A case study using brush-tailed rock-wallabies *Petrogale*  
450 *penicillata*. *Journal of Applied Ecology*, **46**, 842–851.
- 451 Peterman, W.E., Connette, G.M., Semlitsch, R.D. & Eggert, L.S. (2014) Ecological resistance  
452 surfaces predict fine-scale genetic differentiation in a terrestrial woodland salamander.  
453 *Molecular Ecology*, **23**, 2402–2413.
- 454 R Core Team (2014) R: A language and environment for statistical computing. R Foundation for  
455 Statistical Computing, Vienna, Austria. <http://www.R-project.org/>.
- 456 Scrucca, L. (2013) GA: A package for genetic algorithms in R. *Journal of Statistical Software*,  
457 **53**, 1–37.

- 458 Shirk, A.J., Wallin, D.O., Cushman, S.A., Rice, C.G. & Warheit, K.I. (2010) Inferring landscape  
459 effects on gene flow: A new model selection framework. *Molecular Ecology*, **19**, 3603–  
460 3619.
- 461 Sivanandam, S. & Deepa, S. (2007) *Introduction to Genetic Algorithms*. Springer.
- 462 Spear, S.F., Balkenhol, N., Fortin, M.J., McRae, B.H. & Scribner, K. (2010) Use of resistance  
463 surfaces for landscape genetic studies: considerations for parameterization and analysis.  
464 *Molecular Ecology*, **19**, 3576–3591.
- 465 Spear, S.F., Peterson, C.R., Matocq, M.D. & Storfer, A. (2005) Landscape genetics of the  
466 blotched tiger salamander (*Ambystoma tigrinum melanostictum*). *Molecular Ecology*, **14**,  
467 2553–2564.
- 468 Stevens, V.M., Verkenne, C., Vandewoestijne, S., Wesselingh, R.A. & Baguette, M. (2006)  
469 Gene flow and functional connectivity in the natterjack toad. *Molecular Ecology*, **15**,  
470 2333–2344.
- 471 Storfer, A., Murphy, M.A., Evans, J.S., Goldberg, C.S., Robinson, S., Spear, S.F., Dezzani, R.,  
472 Delmelle, E., Vierling, L. & Waits, L.P. (2007) Putting the 'landscape' in landscape  
473 genetics. *Heredity*, **98**, 128–142.
- 474 Storfer, A., Murphy, M.A., Spear, S.F., Holderegger, R. & Waits, L.P. (2010) Landscape  
475 genetics: where are we now? *Molecular Ecology*, **19**, 3496–3514.
- 476 van Etten, J. (2014) gdistance: distances and routes on geographical grids. R package version  
477 1.1-5. <http://CRAN.R-project.org/package=gdistance>.
- 478 Wang, Y.-H., Yang, K.-C., Bridgman, C.L. & Lin, L.-K. (2008) Habitat suitability modelling to  
479 correlate gene flow with landscape connectivity. *Landscape Ecology*, **23**, 989–1000.
- 480 Wickham, H. (2009) *ggplot2: Elegant graphics for data analysis*. Springer, New York.

481 **Table 1.** Functions of the ResistanceGA package

<b>Function</b>	<b>Returned objects</b>	<b>Description</b>
Combine_Surfaces	R raster object	Combine multiple resistance surfaces into a new composite surface based on specified parameters
CS.prep	Named list	This function prepares and bundles the inputs necessary to run CIRCUITSCAPE
Diagnostic.Plots	.tif file	This function returns a multipanel figure including the histogram of residuals and qqplot from a fitted mixed effects model
GA.prep	Named list	This function prepares and compiles the objects and commands needed to execute the genetic algorithm
gdist.prep	Named list	This function prepares and bundles the inputs necessary to run gdistance
Grid.Search	contour plot, R data object	For a single continuous surface, this function can be used to visualize the AICc response surface
Lower	R data object	Convenience function to obtain the lower half of a square distance matrix
MLPE.lmm	lmer object	Fits a maximum likelihood population effects mixed effects model in lme4
MS_optim	GA object, diagnostic plots, .txt summary file	This is a wrapper function for simultaneously optimizing multiple surfaces.
Plot.trans	ggplot object	Implements and plots a continuous surface transformation
Resistance.opt_multi	AICc value	This is the function passed to ga to optimize multiple resistance surfaces simultaneously
Resistance.opt_single	AICc value	This is the function passed to ga to optimize resistance surfaces in isolation
Resistance.tran	R raster object, .asc file	Applies specified transformation to continuous resistance surface and returns a raster object. Optionally, a .asc file can be exported
Run_CS	R raster object or R data object	This function executes CIRCUITSCAPE from R and returns either the lower half of the pairwise resistance distance matrix or the cumulative current map produced by CIRCUITSCAPE

Run_gdistance	costDistance matrix object from gdistance	This function executes gdistance and returns the costDistance matrix object
SS_optim	.tif files, .csv summary tables	This is a wrapper function for optimizing surfaces in isolation. All surfaces in a common directory will be optimized in turn, and numerous summary tables of optimized parameters and AICc values are produced

---

483 **Table 2.** Arguments of the required preparation functions and their default settings. Only one of  
 484 either `CS.prep` or `gdist.prep` needs to be run, depending upon whether optimization will use  
 485 `CIRCUITSCAPE` or `gdistance`.

Function	Arguments	Defaults	Comments
CS.prep	n.POPS	Must be defined	
	response	NULL	Must be defined to run optimization
	CS_Point.File	Must be defined	
	CS.program	"C:/Program Files/Circuitscape/cs_run.exe"	Default Windows installation location
	Neighbor.Connect	8	
gdist.prep	n.POPS	Must be defined	
	response	NULL	Must be defined to run optimization
	samples	Must be defined	
	transitionFunction	function(x) 1/mean(x)	
	directions	8	
	longlat	FALSE	
GA.prep	ASCII.dir	Must be defined	.asc files should be in their own directory
	Min.Max	"max"	Must be "max" when optimization with ga
	min.cat	0.0001	
	max.cat	2500	
	max.cont	2500	
	cont.shape	NULL	
	pop.mult	15	
	percent.elite	0.05	
	type	"real-valued"	Must be "real-valued"
	pcrossover	0.85	
	pmutation	0.1	
	maxiter	1000	
	run	25	
	keepBest	TRUE	
	population	"gareal_Population"	
selection	"gareal_IsSelection"		
crossover	"gareal_blxCrossover"		

mutation	"gareal_raMutation"
seed	NULL
quiet	FALSE

---

486



487 **Figure 1.** There are eight continuous resistance surface data transformations implemented in  
488 `ResistanceGA`. Prior to transformation, the original continuous resistance surface had values  
489 ranging from 0–10. The shape and magnitude of each transformation are each controlled by a  
490 single parameter. All transformations in the figure have a shape parameter value of 3, and  
491 maximum value parameter of 100. Linear relationships are not explicitly incorporated, but all  
492 monomolecular functions become linear as the shape parameter increases.

493

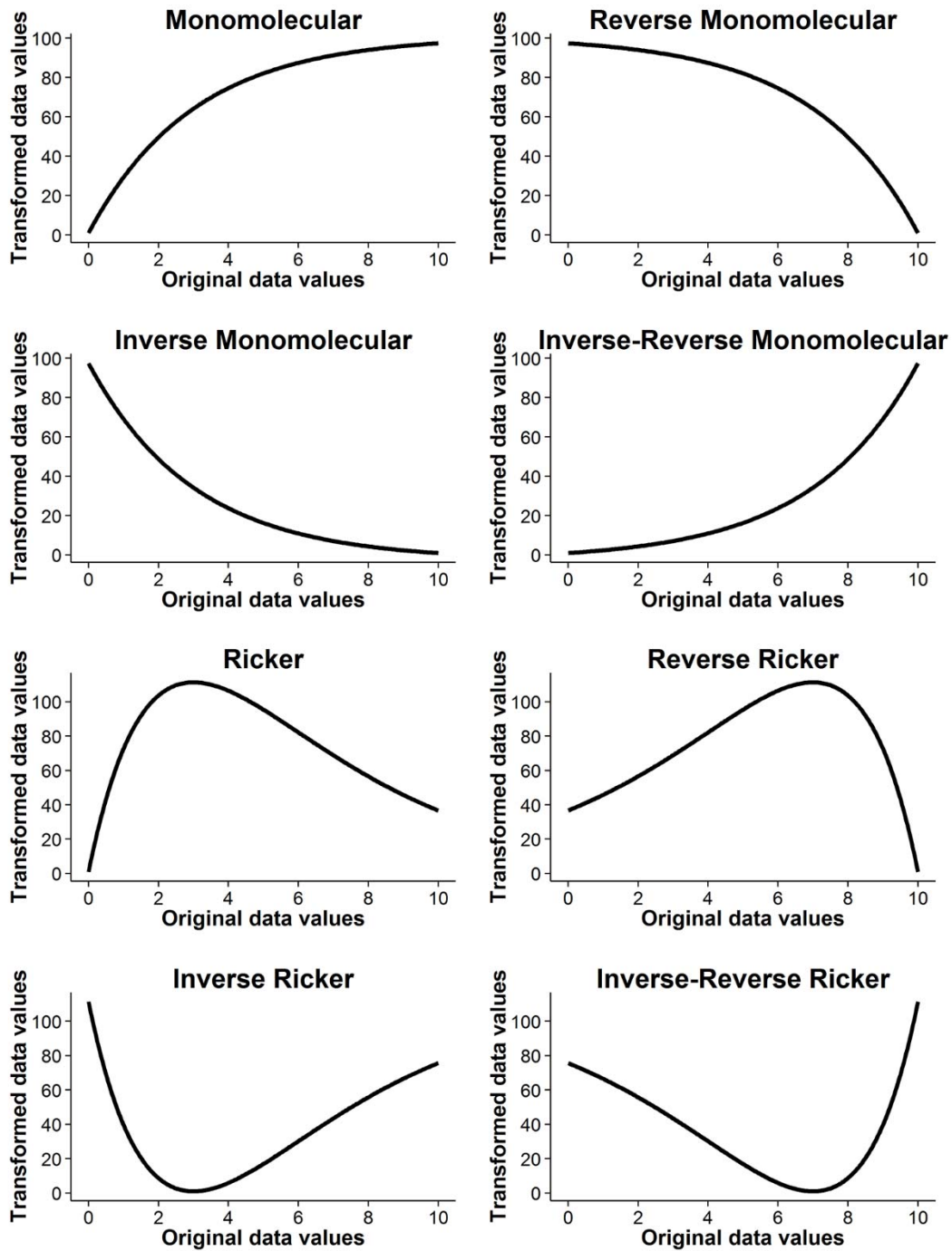
494 **Figure 2.** Flow chart depicting a potential work flow for optimizing resistance surfaces. Analysis  
495 begins by optimizing each resistance surface in isolation. If multiple surfaces are supported (e.g.,  
496  $\Delta AICc \leq 4$ ), these surfaces can be simultaneously optimized to create a composite surface. If  
497 desired, different combinations of the best supported single surfaces can be optimized and  
498 compared. Inputs into the optimization are shown in yellow ovals, optimization steps are blue  
499 rectangles, decision points are white diamonds, and final products from optimization are green  
500 ovals.

501 **Figure 3.** Optimization of a continuous surface that was transformed using a monomolecular  
502 function following Example 1 in the main text (transformation = Monomolecular, shape = 2.0,  
503 maximum = 275). Optimization with `CIRCUITSCAPE` took 147 iterations of the genetic  
504 algorithm and precisely recovered the transformation parameters resulting in perfectly correlated  
505 resistance surfaces.

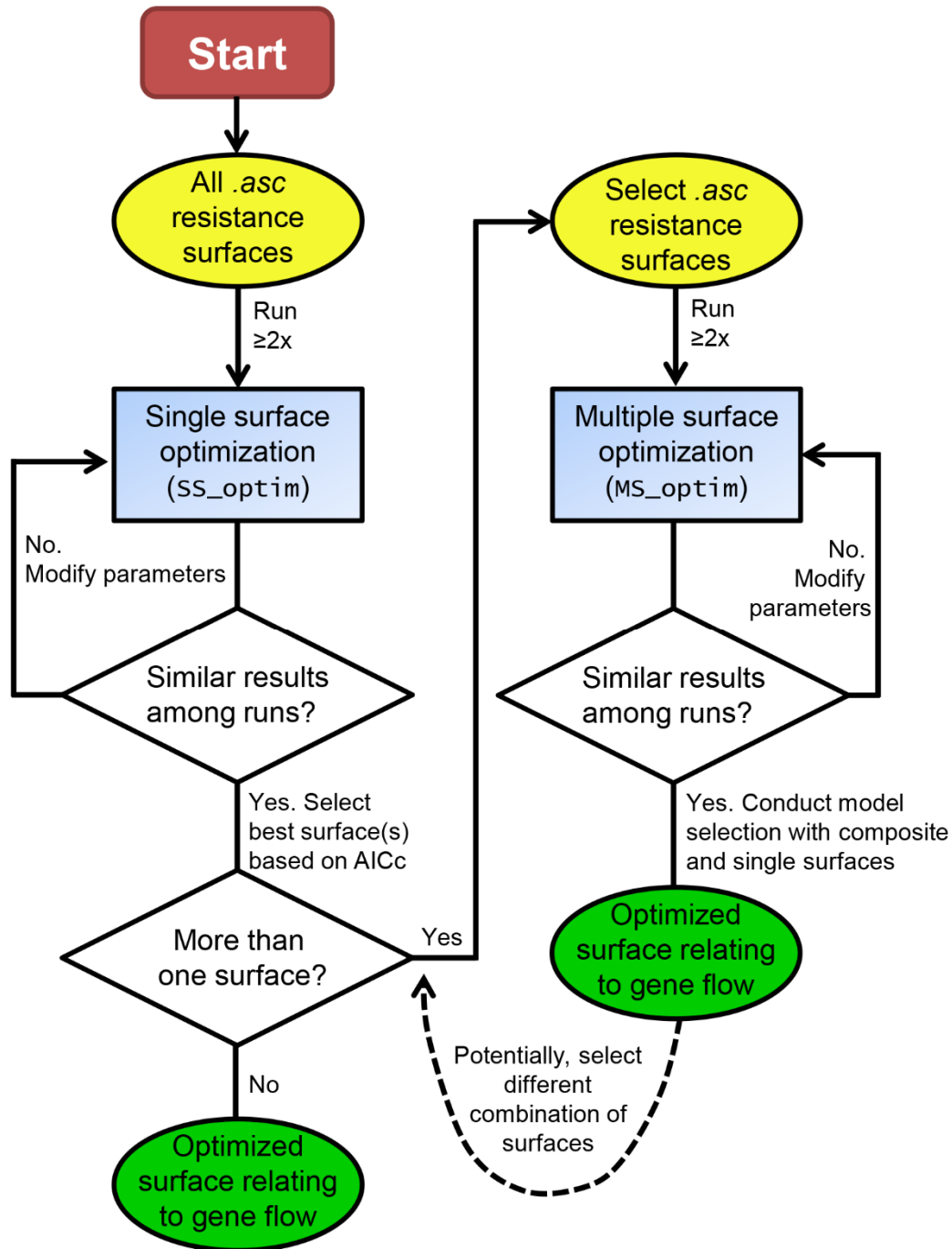
506

507 **Figure 4.** Simultaneous optimization of three resistance surfaces (Example 2 in the main text).  
508 The original categorical, continuous, and feature resistance surfaces are shown on the left and  
509 their actual resistances following transformation/rescaling are shown in the middle. When these  
510 surfaces are added together to create a composite, the minimum resistance value is no longer 1,  
511 so the surface is rescaled to present the relative resistance values (scaled composite resistance).  
512 Optimization using least cost paths took 215 iterations of the genetic algorithm when run in  
513 parallel on 4 cores. The relative resistance values of the optimized surface and the true resistance  
514 surface are perfectly correlated.

515

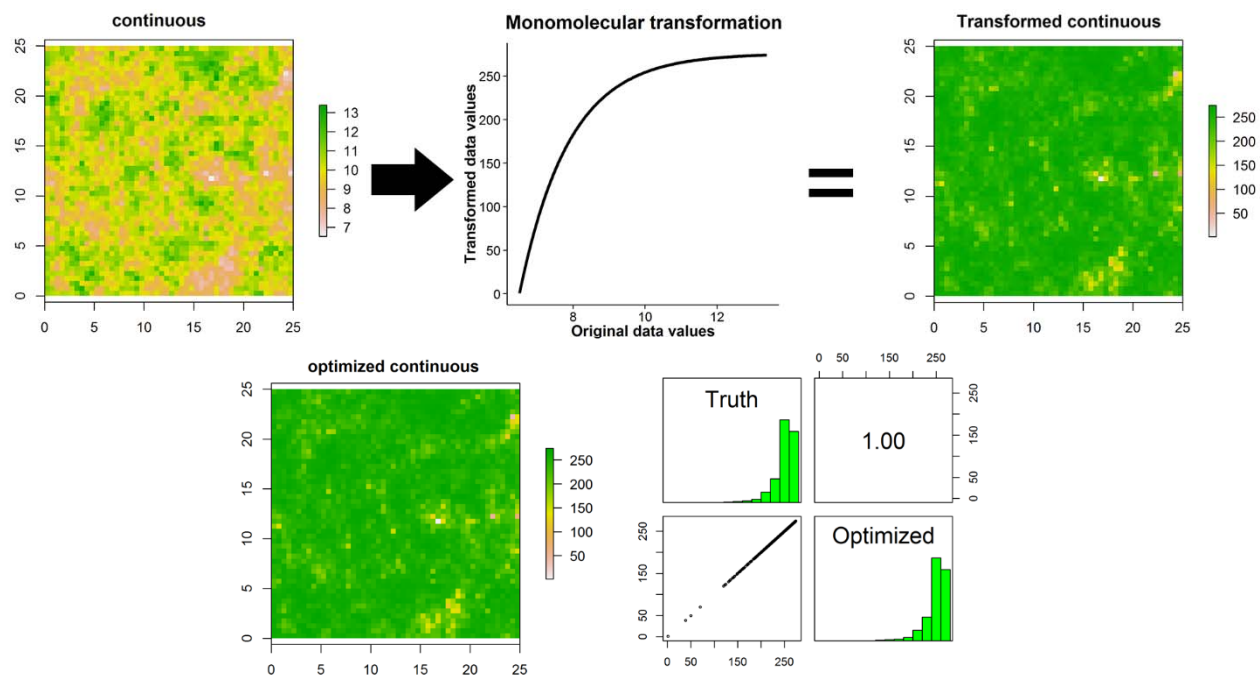


516 Figure 1



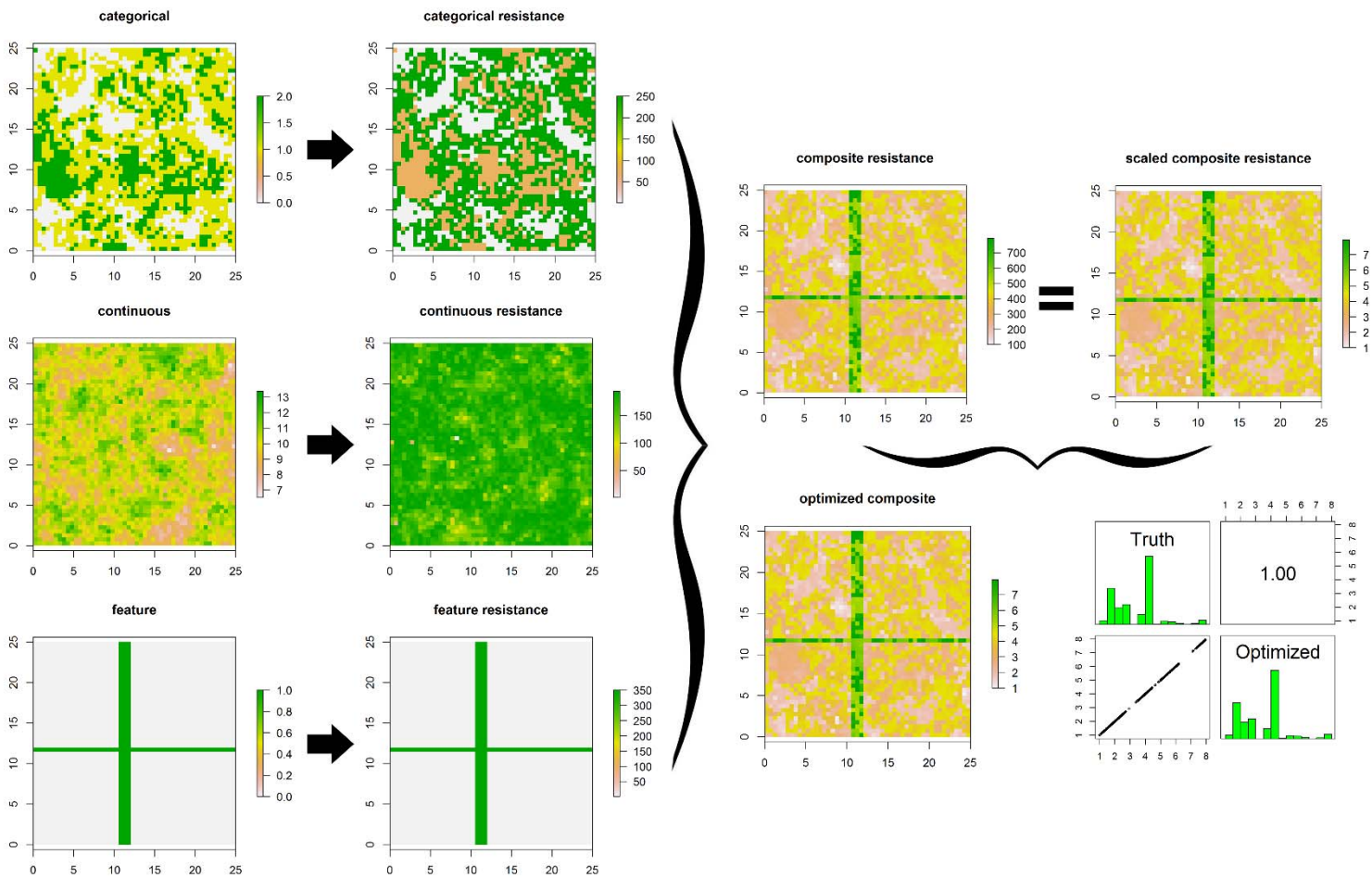
517

518 Figure 2



519

520 Figure 3



521

522 Figure 4