ipmr

init_ipm()

define_kernel()

define_impl()

define_domains()

define_pop_state()

make_ipm()

- Vital rate modelling left outside the package

- Designed to reflect mathematical notation

- Reduces required time and coding knowledge for building IPMs init_ipm()

define_kernel()

define_impl()

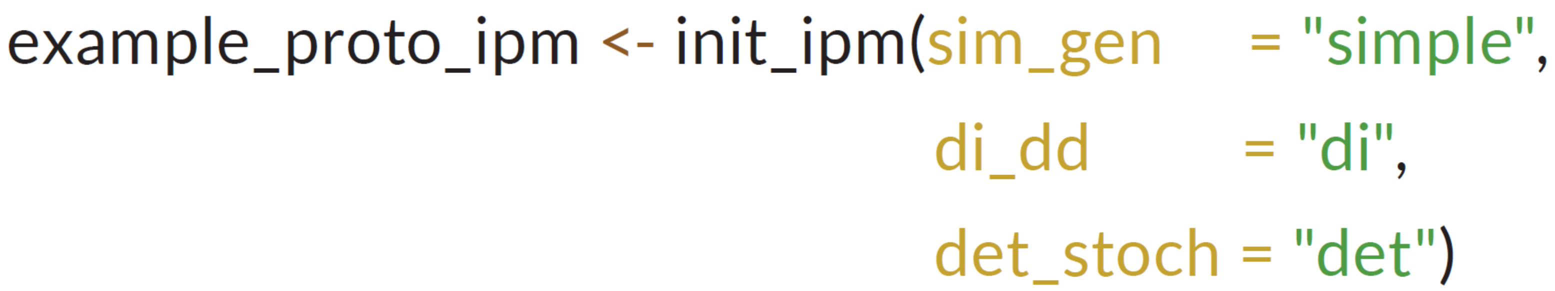
define_domains()

define_pop_state()

make_ipm()

ipmr structure

- Start-up ipmr



```
example_proto_ipm <- init_ipm(sim_gen     = "simple",
                              di_dd       = "di",
                              det_stoch = "det")
```

ipmr structure

init_ipm()

define_kernel()

define_impl()

define_domains()

define_pop_state()

make_ipm()

```
example_proto_ipm <- define_kernel(
   example_proto_ipm,
   name            = "P",
   formula         = surv * Grow,
   surv            = plogis(s_i + s_z * z_1),
   Grow            = dnorm(z_2, mu_G, sd_G),
   mu_G            = G_i + G_z * z_1,
   data_list       = data_list,
   states          = list(c("z"))
)
```

ipmr structure

- Eviction correction

init_ipm()

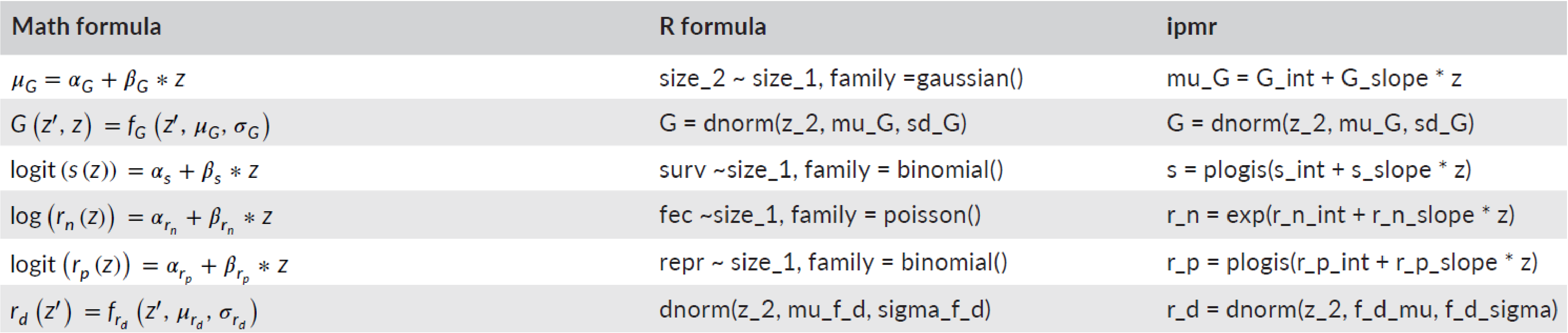
define_kernel()

define_impl()

define_domains()

define_pop_state()

make_ipm()



Math formula	R formula	ipmr
$\mu_G = \alpha_G + \beta_G * z$	size_2 ~ size_1, family =gaussian()	mu_G = G_int + G_slope * z
$G(z', z) = f_G(z', \mu_G, \sigma_G)$	G = dnorm(z_2, mu_G, sd_G)	G = dnorm(z_2, mu_G, sd_G)
$\text{logit}(s(z)) = \alpha_s + \beta_s * z$	surv ~size_1, family = binomial()	s = plogis(s_int + s_slope * z)
$\log(r_n(z)) = \alpha_{r_n} + \beta_{r_n} * z$	fec ~size_1, family = poisson()	r_n = exp(r_n_int + r_n_slope * z)
$\text{logit}(r_p(z)) = \alpha_{r_p} + \beta_{r_p} * z$	repr ~ size_1, family = binomial()	r_p = plogis(r_p_int + r_p_slope * z)
$r_d(z') = f_{r_d}(z', \mu_{r_d}, \sigma_{r_d})$	dnorm(z_2, mu_f_d, sigma_f_d)	r_d = dnorm(z_2, f_d_mu, f_d_sigma)

Levin et al., 2021 – (part of) Table 1

ipmr structure

init_ipm()

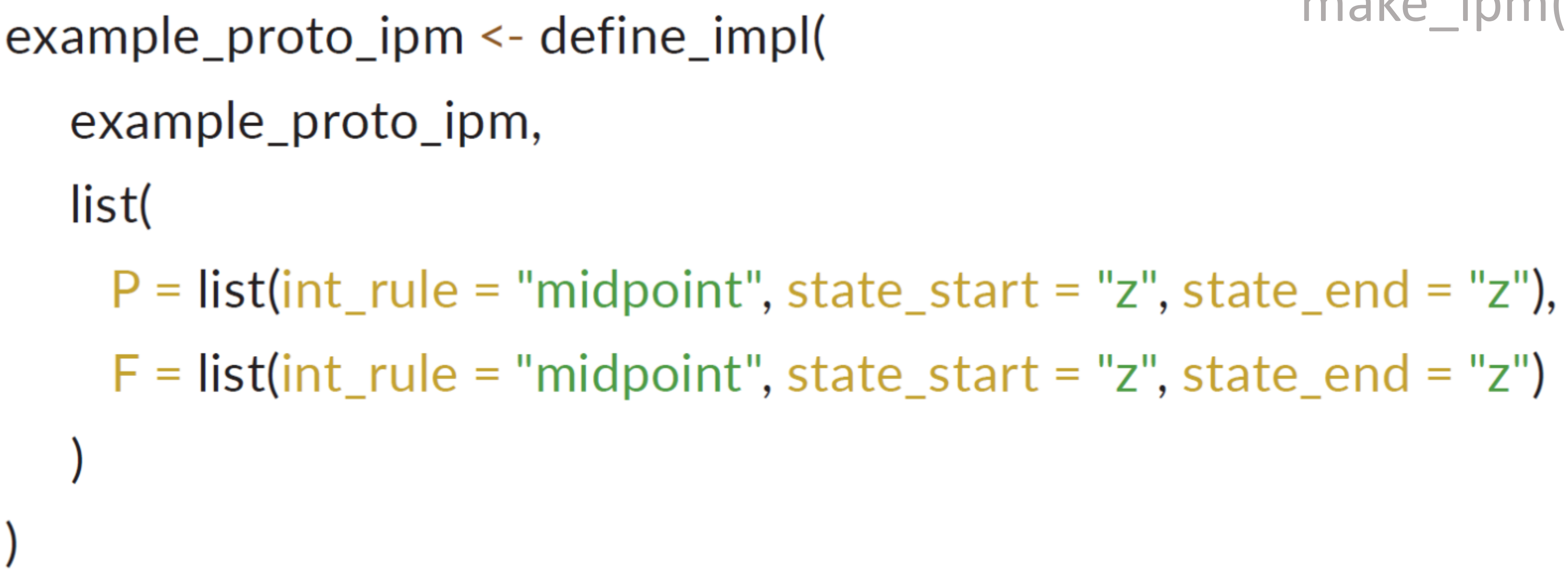
define_kernel()

define_impl()

define_domains()

define_pop_state()

make_ipm()



```
example_proto_ipm <- define_impl(
    example_proto_ipm,
    list(
      P = list(int_rule = "midpoint", state_start = "z", state_end = "z"),
      F = list(int_rule = "midpoint", state_start = "z", state_end = "z")
    )
)
```

ipmr structure

init_ipm()

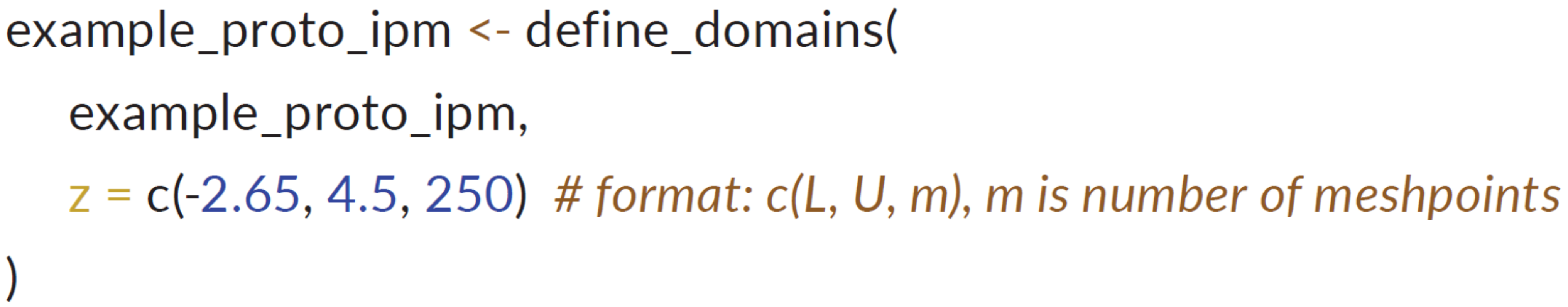
define_kernel()

define_impl()

define_domains()

define_pop_state()

make_ipm()



```
example_proto_ipm <- define_domains(
    example_proto_ipm,
    z = c(-2.65, 4.5, 250)  # format: c(L, U, m), m is number of meshpoints
)
```

ipmr structure

init_ipm()

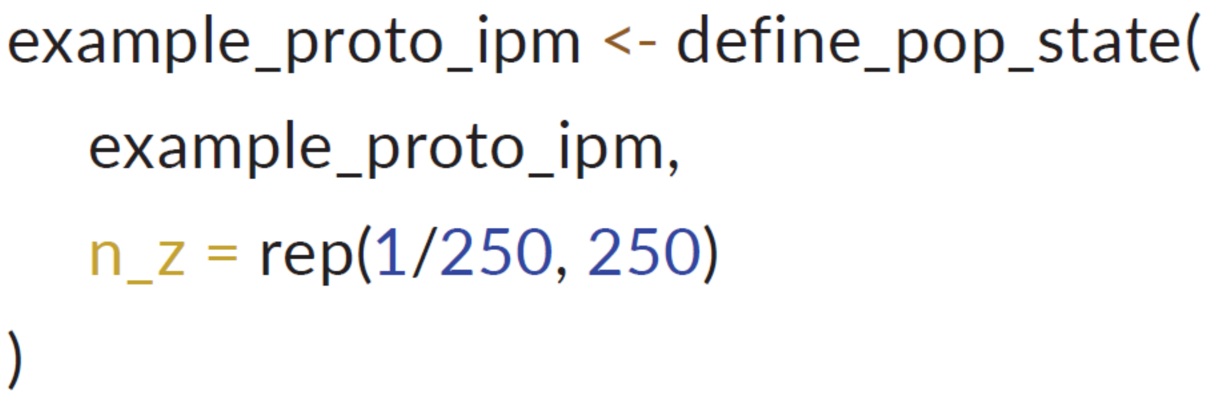
define_kernel()

define_impl()

define_domains()

define_pop_state()

make_ipm()



```
example_proto_ipm <- define_pop_state(
    example_proto_ipm,
    n_z = rep(1/250, 250)
)
```

ipmr structure

```
example_ipm <- make_ipm(example_proto_ipm)
```

init_ipm()

define_kernel()

define_impl()

define_domains()

define_pop_state()

make_ipm()