> model.fit <- iter.func(ss = 2000, sim = 100)

DIAGNOSTIC(S) FROM PARSER:

Warning (non-fatal): assignment operator <- deprecated in the Stan language; use = instead.

In file included from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/prim/mat.hpp:36:0,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/rev/mat.hpp:8,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math.hpp:4,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/src/stan/model/model\_header.hpp:4,

from filee0843b44cb1.cpp:8:

C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/prim/mat/err/check\_positive\_ordered.hpp: In function 'bool stan::math::check\_positive\_ordered(const char\*, const char\*, const Eigen::Matrix<Scalar, -1, 1>&)':

C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/prim/mat/err/check\_positive\_ordered.hpp:39:67: warning: typedef 'size\_type' locally defined but not used [-Wunused-local-typedefs]

typedef typename index\_type<Matrix<T\_y, Dynamic, 1> >::type size\_type;

^

In file included from C:/Users/huiying Chua/Documents/R/win-library/3.3/BH/include/boost/multi\_array/base.hpp:28:0,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/BH/include/boost/multi\_array.hpp:21,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/BH/include/boost/numeric/odeint/util/multi\_array\_adaption.hpp:29,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/BH/include/boost/numeric/odeint.hpp:61,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/prim/arr/functor/integrate\_ode\_rk45.hpp:13,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/prim/arr.hpp:33,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/prim/mat.hpp:232,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/rev/mat.hpp:8,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math.hpp:4,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/src/stan/model/model\_header.hpp:4,

from filee0843b44cb1.cpp:8:

C:/Users/huiying Chua/Documents/R/win-library/3.3/BH/include/boost/multi\_array/concept\_checks.hpp: In static member function 'static void boost::multi\_array\_concepts::detail::idgen\_helper<N>::call(Array&, const IdxGen&, Call\_Type)':

C:/Users/huiying Chua/Documents/R/win-library/3.3/BH/include/boost/multi\_array/concept\_checks.hpp:42:43: warning: typedef 'index\_range' locally defined but not used [-Wunused-local-typedefs]

typedef typename Array::index\_range index\_range;

^

C:/Users/huiying Chua/Documents/R/win-library/3.3/BH/include/boost/multi\_array/concept\_checks.hpp:43:37: warning: typedef 'index' locally defined but not used [-Wunused-local-typedefs]

typedef typename Array::index index;

^

C:/Users/huiying Chua/Documents/R/win-library/3.3/BH/include/boost/multi\_array/concept\_checks.hpp: In static member function 'static void boost::multi\_array\_concepts::detail::idgen\_helper<0ull>::call(Array&, const IdxGen&, Call\_Type)':

C:/Users/huiying Chua/Documents/R/win-library/3.3/BH/include/boost/multi\_array/concept\_checks.hpp:53:43: warning: typedef 'index\_range' locally defined but not used [-Wunused-local-typedefs]

typedef typename Array::index\_range index\_range;

^

C:/Users/huiying Chua/Documents/R/win-library/3.3/BH/include/boost/multi\_array/concept\_checks.hpp:54:37: warning: typedef 'index' locally defined but not used [-Wunused-local-typedefs]

typedef typename Array::index index;

^

In file included from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/rev/core.hpp:42:0,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/rev/mat.hpp:4,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math.hpp:4,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/src/stan/model/model\_header.hpp:4,

from filee0843b44cb1.cpp:8:

C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/rev/core/set\_zero\_all\_adjoints.hpp: At global scope:

C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/rev/core/set\_zero\_all\_adjoints.hpp:14:17: warning: 'void stan::math::set\_zero\_all\_adjoints()' defined but not used [-Wunused-function]

static void set\_zero\_all\_adjoints() {

^

g++.exe: error: Chua/Documents/R/win-library/3.3/StanHeaders/libs/x64: No such file or directory

ERROR(s) during compilation: source code errors or compiler configuration errors!

Program source:

1:

2: // includes from the plugin

3:

4:

5: // user includes

6: #define STAN\_\_SERVICES\_\_COMMAND\_HPP// Code generated by Stan version 2.10

7:

8: #include <stan/model/model\_header.hpp>

9:

10: namespace modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace {

11:

12: using std::istream;

13: using std::string;

14: using std::stringstream;

15: using std::vector;

16: using stan::io::dump;

17: using stan::math::lgamma;

18: using stan::model::prob\_grad;

19: using namespace stan::math;

20:

21: typedef Eigen::Matrix<double,Eigen::Dynamic,1> vector\_d;

22: typedef Eigen::Matrix<double,1,Eigen::Dynamic> row\_vector\_d;

23: typedef Eigen::Matrix<double,Eigen::Dynamic,Eigen::Dynamic> matrix\_d;

24:

25: static int current\_statement\_begin\_\_;

26:

27: class modele084e5e7a89\_2016\_06\_14\_tnd\_misclass : public prob\_grad {

28: private:

29: int N;

30: vector<int> vaccine;

31: vector<int> delay;

32: vector<int> true\_flu;

33: vector<int> obs\_flu;

34: public:

35: modele084e5e7a89\_2016\_06\_14\_tnd\_misclass(stan::io::var\_context& context\_\_,

36: std::ostream\* pstream\_\_ = 0)

37: : prob\_grad(0) {

38: current\_statement\_begin\_\_ = -1;

39:

40: static const char\* function\_\_ = "modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass";

41: (void) function\_\_; // dummy call to supress warning

42: size\_t pos\_\_;

43: (void) pos\_\_; // dummy call to supress warning

44: std::vector<int> vals\_i\_\_;

45: std::vector<double> vals\_r\_\_;

46: context\_\_.validate\_dims("data initialization", "N", "int", context\_\_.to\_vec());

47: N = int(0);

48: vals\_i\_\_ = context\_\_.vals\_i("N");

49: pos\_\_ = 0;

50: N = vals\_i\_\_[pos\_\_++];

51: context\_\_.validate\_dims("data initialization", "vaccine", "int", context\_\_.to\_vec(N));

52: validate\_non\_negative\_index("vaccine", "N", N);

53: vaccine = std::vector<int>(N,int(0));

54: vals\_i\_\_ = context\_\_.vals\_i("vaccine");

55: pos\_\_ = 0;

56: size\_t vaccine\_limit\_0\_\_ = N;

57: for (size\_t i\_0\_\_ = 0; i\_0\_\_ < vaccine\_limit\_0\_\_; ++i\_0\_\_) {

58: vaccine[i\_0\_\_] = vals\_i\_\_[pos\_\_++];

59: }

60: context\_\_.validate\_dims("data initialization", "delay", "int", context\_\_.to\_vec(N));

61: validate\_non\_negative\_index("delay", "N", N);

62: delay = std::vector<int>(N,int(0));

63: vals\_i\_\_ = context\_\_.vals\_i("delay");

64: pos\_\_ = 0;

65: size\_t delay\_limit\_0\_\_ = N;

66: for (size\_t i\_0\_\_ = 0; i\_0\_\_ < delay\_limit\_0\_\_; ++i\_0\_\_) {

67: delay[i\_0\_\_] = vals\_i\_\_[pos\_\_++];

68: }

69: context\_\_.validate\_dims("data initialization", "true\_flu", "int", context\_\_.to\_vec(N));

70: validate\_non\_negative\_index("true\_flu", "N", N);

71: true\_flu = std::vector<int>(N,int(0));

72: vals\_i\_\_ = context\_\_.vals\_i("true\_flu");

73: pos\_\_ = 0;

74: size\_t true\_flu\_limit\_0\_\_ = N;

75: for (size\_t i\_0\_\_ = 0; i\_0\_\_ < true\_flu\_limit\_0\_\_; ++i\_0\_\_) {

76: true\_flu[i\_0\_\_] = vals\_i\_\_[pos\_\_++];

77: }

78: context\_\_.validate\_dims("data initialization", "obs\_flu", "int", context\_\_.to\_vec(N));

79: validate\_non\_negative\_index("obs\_flu", "N", N);

80: obs\_flu = std::vector<int>(N,int(0));

81: vals\_i\_\_ = context\_\_.vals\_i("obs\_flu");

82: pos\_\_ = 0;

83: size\_t obs\_flu\_limit\_0\_\_ = N;

84: for (size\_t i\_0\_\_ = 0; i\_0\_\_ < obs\_flu\_limit\_0\_\_; ++i\_0\_\_) {

85: obs\_flu[i\_0\_\_] = vals\_i\_\_[pos\_\_++];

86: }

87:

88: // validate data

89:

90: double DUMMY\_VAR\_\_(std::numeric\_limits<double>::quiet\_NaN());

91: (void) DUMMY\_VAR\_\_; // suppress unused var warning

92:

93:

94: // initialize transformed variables to avoid seg fault on val access

95:

96: try {

97: } catch (const std::exception& e) {

98: stan::lang::rethrow\_located(e,current\_statement\_begin\_\_);

99: // Next line prevents compiler griping about no return

100: throw std::runtime\_error("\*\*\* IF YOU SEE THIS, PLEASE REPORT A BUG \*\*\*");

101: }

102:

103: // validate transformed data

104:

105: // set parameter ranges

106: num\_params\_r\_\_ = 0U;

107: param\_ranges\_i\_\_.clear();

108: ++num\_params\_r\_\_;

109: ++num\_params\_r\_\_;

110: ++num\_params\_r\_\_;

111: ++num\_params\_r\_\_;

112: ++num\_params\_r\_\_;

113: }

114:

115: ~modele084e5e7a89\_2016\_06\_14\_tnd\_misclass() { }

116:

117:

118: void transform\_inits(const stan::io::var\_context& context\_\_,

119: std::vector<int>& params\_i\_\_,

120: std::vector<double>& params\_r\_\_,

121: std::ostream\* pstream\_\_) const {

122: stan::io::writer<double> writer\_\_(params\_r\_\_,params\_i\_\_);

123: size\_t pos\_\_;

124: (void) pos\_\_; // dummy call to supress warning

125: std::vector<double> vals\_r\_\_;

126: std::vector<int> vals\_i\_\_;

127:

128: if (!(context\_\_.contains\_r("pi0")))

129: throw std::runtime\_error("variable pi0 missing");

130: vals\_r\_\_ = context\_\_.vals\_r("pi0");

131: pos\_\_ = 0U;

132: context\_\_.validate\_dims("initialization", "pi0", "double", context\_\_.to\_vec());

133: double pi0(0);

134: pi0 = vals\_r\_\_[pos\_\_++];

135: try {

136: writer\_\_.scalar\_lub\_unconstrain(0,1,pi0);

137: } catch (const std::exception& e) {

138: throw std::runtime\_error(std::string("Error transforming variable pi0: ") + e.what());

139: }

140:

141: if (!(context\_\_.contains\_r("pi1")))

142: throw std::runtime\_error("variable pi1 missing");

143: vals\_r\_\_ = context\_\_.vals\_r("pi1");

144: pos\_\_ = 0U;

145: context\_\_.validate\_dims("initialization", "pi1", "double", context\_\_.to\_vec());

146: double pi1(0);

147: pi1 = vals\_r\_\_[pos\_\_++];

148: try {

149: writer\_\_.scalar\_lub\_unconstrain(0,1,pi1);

150: } catch (const std::exception& e) {

151: throw std::runtime\_error(std::string("Error transforming variable pi1: ") + e.what());

152: }

153:

154: if (!(context\_\_.contains\_r("VE")))

155: throw std::runtime\_error("variable VE missing");

156: vals\_r\_\_ = context\_\_.vals\_r("VE");

157: pos\_\_ = 0U;

158: context\_\_.validate\_dims("initialization", "VE", "double", context\_\_.to\_vec());

159: double VE(0);

160: VE = vals\_r\_\_[pos\_\_++];

161: try {

162: writer\_\_.scalar\_lub\_unconstrain(0,1,VE);

163: } catch (const std::exception& e) {

164: throw std::runtime\_error(std::string("Error transforming variable VE: ") + e.what());

165: }

166:

167: if (!(context\_\_.contains\_r("sens0")))

168: throw std::runtime\_error("variable sens0 missing");

169: vals\_r\_\_ = context\_\_.vals\_r("sens0");

170: pos\_\_ = 0U;

171: context\_\_.validate\_dims("initialization", "sens0", "double", context\_\_.to\_vec());

172: double sens0(0);

173: sens0 = vals\_r\_\_[pos\_\_++];

174: try {

175: writer\_\_.scalar\_lub\_unconstrain(0,1,sens0);

176: } catch (const std::exception& e) {

177: throw std::runtime\_error(std::string("Error transforming variable sens0: ") + e.what());

178: }

179:

180: if (!(context\_\_.contains\_r("sens1")))

181: throw std::runtime\_error("variable sens1 missing");

182: vals\_r\_\_ = context\_\_.vals\_r("sens1");

183: pos\_\_ = 0U;

184: context\_\_.validate\_dims("initialization", "sens1", "double", context\_\_.to\_vec());

185: double sens1(0);

186: sens1 = vals\_r\_\_[pos\_\_++];

187: try {

188: writer\_\_.scalar\_lub\_unconstrain(0,1,sens1);

189: } catch (const std::exception& e) {

190: throw std::runtime\_error(std::string("Error transforming variable sens1: ") + e.what());

191: }

192:

193: params\_r\_\_ = writer\_\_.data\_r();

194: params\_i\_\_ = writer\_\_.data\_i();

195: }

196:

197: void transform\_inits(const stan::io::var\_context& context,

198: Eigen::Matrix<double,Eigen::Dynamic,1>& params\_r,

199: std::ostream\* pstream\_\_) const {

200: std::vector<double> params\_r\_vec;

201: std::vector<int> params\_i\_vec;

202: transform\_inits(context, params\_i\_vec, params\_r\_vec, pstream\_\_);

203: params\_r.resize(params\_r\_vec.size());

204: for (int i = 0; i < params\_r.size(); ++i)

205: params\_r(i) = params\_r\_vec[i];

206: }

207:

208:

209: template <bool propto\_\_, bool jacobian\_\_, typename T\_\_>

210: T\_\_ log\_prob(vector<T\_\_>& params\_r\_\_,

211: vector<int>& params\_i\_\_,

212: std::ostream\* pstream\_\_ = 0) const {

213:

214: T\_\_ DUMMY\_VAR\_\_(std::numeric\_limits<double>::quiet\_NaN());

215: (void) DUMMY\_VAR\_\_; // suppress unused var warning

216:

217: T\_\_ lp\_\_(0.0);

218: stan::math::accumulator<T\_\_> lp\_accum\_\_;

219:

220: // model parameters

221: stan::io::reader<T\_\_> in\_\_(params\_r\_\_,params\_i\_\_);

222:

223: T\_\_ pi0;

224: (void) pi0; // dummy to suppress unused var warning

225: if (jacobian\_\_)

226: pi0 = in\_\_.scalar\_lub\_constrain(0,1,lp\_\_);

227: else

228: pi0 = in\_\_.scalar\_lub\_constrain(0,1);

229:

230: T\_\_ pi1;

231: (void) pi1; // dummy to suppress unused var warning

232: if (jacobian\_\_)

233: pi1 = in\_\_.scalar\_lub\_constrain(0,1,lp\_\_);

234: else

235: pi1 = in\_\_.scalar\_lub\_constrain(0,1);

236:

237: T\_\_ VE;

238: (void) VE; // dummy to suppress unused var warning

239: if (jacobian\_\_)

240: VE = in\_\_.scalar\_lub\_constrain(0,1,lp\_\_);

241: else

242: VE = in\_\_.scalar\_lub\_constrain(0,1);

243:

244: T\_\_ sens0;

245: (void) sens0; // dummy to suppress unused var warning

246: if (jacobian\_\_)

247: sens0 = in\_\_.scalar\_lub\_constrain(0,1,lp\_\_);

248: else

249: sens0 = in\_\_.scalar\_lub\_constrain(0,1);

250:

251: T\_\_ sens1;

252: (void) sens1; // dummy to suppress unused var warning

253: if (jacobian\_\_)

254: sens1 = in\_\_.scalar\_lub\_constrain(0,1,lp\_\_);

255: else

256: sens1 = in\_\_.scalar\_lub\_constrain(0,1);

257:

258:

259: // transformed parameters

260: vector<T\_\_> p(N);

261:

262: // initialize transformed variables to avoid seg fault on val access

263: stan::math::fill(p,DUMMY\_VAR\_\_);

264:

265: try {

266: current\_statement\_begin\_\_ = 27;

267: for (int iin = 1; iin <= N; ++iin) {

268: current\_statement\_begin\_\_ = 28;

269: stan::math::assign(get\_base1\_lhs(p,iin,"p",1), ((((1 - get\_base1(delay,iin,"delay",1)) \* (((1 - get\_base1(vaccine,iin,"vaccine",1)) \* pi0) + ((get\_base1(vaccine,iin,"vaccine",1) \* pi0) \* (1 - VE)))) + (get\_base1(delay,iin,"delay",1) \* (((1 - get\_base1(vaccine,iin,"vaccine",1)) \* pi1) + ((get\_base1(vaccine,iin,"vaccine",1) \* pi1) \* (1 - VE))))) \* (((1 - get\_base1(delay,iin,"delay",1)) \* sens0) + (get\_base1(delay,iin,"delay",1) \* sens1))));

270: }

271: } catch (const std::exception& e) {

272: stan::lang::rethrow\_located(e,current\_statement\_begin\_\_);

273: // Next line prevents compiler griping about no return

274: throw std::runtime\_error("\*\*\* IF YOU SEE THIS, PLEASE REPORT A BUG \*\*\*");

275: }

276:

277: // validate transformed parameters

278: for (int i0\_\_ = 0; i0\_\_ < N; ++i0\_\_) {

279: if (stan::math::is\_uninitialized(p[i0\_\_])) {

280: std::stringstream msg\_\_;

281: msg\_\_ << "Undefined transformed parameter: p" << '[' << i0\_\_ << ']';

282: throw std::runtime\_error(msg\_\_.str());

283: }

284: }

285:

286: const char\* function\_\_ = "validate transformed params";

287: (void) function\_\_; // dummy to suppress unused var warning

288: for (int k0\_\_ = 0; k0\_\_ < N; ++k0\_\_) {

289: check\_greater\_or\_equal(function\_\_,"p[k0\_\_]",p[k0\_\_],0);

290: check\_less\_or\_equal(function\_\_,"p[k0\_\_]",p[k0\_\_],1);

291: }

292:

293: // model body

294: try {

295: current\_statement\_begin\_\_ = 38;

296: lp\_accum\_\_.add(uniform\_log<propto\_\_>(pi0, 0, 1));

297: current\_statement\_begin\_\_ = 39;

298: lp\_accum\_\_.add(uniform\_log<propto\_\_>(pi1, 0, 1));

299: current\_statement\_begin\_\_ = 40;

300: lp\_accum\_\_.add(uniform\_log<propto\_\_>(VE, 0, 1));

301: current\_statement\_begin\_\_ = 41;

302: lp\_accum\_\_.add(beta\_log<propto\_\_>(sens0, 9.5, 0.5));

303: current\_statement\_begin\_\_ = 42;

304: lp\_accum\_\_.add(beta\_log<propto\_\_>(sens1, 4, 6));

305: current\_statement\_begin\_\_ = 45;

306: for (int iin = 1; iin <= N; ++iin) {

307: current\_statement\_begin\_\_ = 46;

308: lp\_accum\_\_.add(binomial\_log<propto\_\_>(get\_base1(obs\_flu,iin,"obs\_flu",1), 1, get\_base1(p,iin,"p",1)));

309: }

310: } catch (const std::exception& e) {

311: stan::lang::rethrow\_located(e,current\_statement\_begin\_\_);

312: // Next line prevents compiler griping about no return

313: throw std::runtime\_error("\*\*\* IF YOU SEE THIS, PLEASE REPORT A BUG \*\*\*");

314: }

315:

316: lp\_accum\_\_.add(lp\_\_);

317: return lp\_accum\_\_.sum();

318:

319: } // log\_prob()

320:

321: template <bool propto, bool jacobian, typename T\_>

322: T\_ log\_prob(Eigen::Matrix<T\_,Eigen::Dynamic,1>& params\_r,

323: std::ostream\* pstream = 0) const {

324: std::vector<T\_> vec\_params\_r;

325: vec\_params\_r.reserve(params\_r.size());

326: for (int i = 0; i < params\_r.size(); ++i)

327: vec\_params\_r.push\_back(params\_r(i));

328: std::vector<int> vec\_params\_i;

329: return log\_prob<propto,jacobian,T\_>(vec\_params\_r, vec\_params\_i, pstream);

330: }

331:

332:

333: void get\_param\_names(std::vector<std::string>& names\_\_) const {

334: names\_\_.resize(0);

335: names\_\_.push\_back("pi0");

336: names\_\_.push\_back("pi1");

337: names\_\_.push\_back("VE");

338: names\_\_.push\_back("sens0");

339: names\_\_.push\_back("sens1");

340: names\_\_.push\_back("p");

341: }

342:

343:

344: void get\_dims(std::vector<std::vector<size\_t> >& dimss\_\_) const {

345: dimss\_\_.resize(0);

346: std::vector<size\_t> dims\_\_;

347: dims\_\_.resize(0);

348: dimss\_\_.push\_back(dims\_\_);

349: dims\_\_.resize(0);

350: dimss\_\_.push\_back(dims\_\_);

351: dims\_\_.resize(0);

352: dimss\_\_.push\_back(dims\_\_);

353: dims\_\_.resize(0);

354: dimss\_\_.push\_back(dims\_\_);

355: dims\_\_.resize(0);

356: dimss\_\_.push\_back(dims\_\_);

357: dims\_\_.resize(0);

358: dims\_\_.push\_back(N);

359: dimss\_\_.push\_back(dims\_\_);

360: }

361:

362: template <typename RNG>

363: void write\_array(RNG& base\_rng\_\_,

364: std::vector<double>& params\_r\_\_,

365: std::vector<int>& params\_i\_\_,

366: std::vector<double>& vars\_\_,

367: bool include\_tparams\_\_ = true,

368: bool include\_gqs\_\_ = true,

369: std::ostream\* pstream\_\_ = 0) const {

370: vars\_\_.resize(0);

371: stan::io::reader<double> in\_\_(params\_r\_\_,params\_i\_\_);

372: static const char\* function\_\_ = "modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::write\_array";

373: (void) function\_\_; // dummy call to supress warning

374: // read-transform, write parameters

375: double pi0 = in\_\_.scalar\_lub\_constrain(0,1);

376: double pi1 = in\_\_.scalar\_lub\_constrain(0,1);

377: double VE = in\_\_.scalar\_lub\_constrain(0,1);

378: double sens0 = in\_\_.scalar\_lub\_constrain(0,1);

379: double sens1 = in\_\_.scalar\_lub\_constrain(0,1);

380: vars\_\_.push\_back(pi0);

381: vars\_\_.push\_back(pi1);

382: vars\_\_.push\_back(VE);

383: vars\_\_.push\_back(sens0);

384: vars\_\_.push\_back(sens1);

385:

386: if (!include\_tparams\_\_) return;

387: // declare and define transformed parameters

388: double lp\_\_ = 0.0;

389: (void) lp\_\_; // dummy call to supress warning

390: stan::math::accumulator<double> lp\_accum\_\_;

391:

392: vector<double> p(N, 0.0);

393:

394: try {

395: current\_statement\_begin\_\_ = 27;

396: for (int iin = 1; iin <= N; ++iin) {

397: current\_statement\_begin\_\_ = 28;

398: stan::math::assign(get\_base1\_lhs(p,iin,"p",1), ((((1 - get\_base1(delay,iin,"delay",1)) \* (((1 - get\_base1(vaccine,iin,"vaccine",1)) \* pi0) + ((get\_base1(vaccine,iin,"vaccine",1) \* pi0) \* (1 - VE)))) + (get\_base1(delay,iin,"delay",1) \* (((1 - get\_base1(vaccine,iin,"vaccine",1)) \* pi1) + ((get\_base1(vaccine,iin,"vaccine",1) \* pi1) \* (1 - VE))))) \* (((1 - get\_base1(delay,iin,"delay",1)) \* sens0) + (get\_base1(delay,iin,"delay",1) \* sens1))));

399: }

400: } catch (const std::exception& e) {

401: stan::lang::rethrow\_located(e,current\_statement\_begin\_\_);

402: // Next line prevents compiler griping about no return

403: throw std::runtime\_error("\*\*\* IF YOU SEE THIS, PLEASE REPORT A BUG \*\*\*");

404: }

405:

406: // validate transformed parameters

407: for (int k0\_\_ = 0; k0\_\_ < N; ++k0\_\_) {

408: check\_greater\_or\_equal(function\_\_,"p[k0\_\_]",p[k0\_\_],0);

409: check\_less\_or\_equal(function\_\_,"p[k0\_\_]",p[k0\_\_],1);

410: }

411:

412: // write transformed parameters

413: for (int k\_0\_\_ = 0; k\_0\_\_ < N; ++k\_0\_\_) {

414: vars\_\_.push\_back(p[k\_0\_\_]);

415: }

416:

417: if (!include\_gqs\_\_) return;

418: // declare and define generated quantities

419:

420: double DUMMY\_VAR\_\_(std::numeric\_limits<double>::quiet\_NaN());

421: (void) DUMMY\_VAR\_\_; // suppress unused var warning

422:

423:

424: // initialize transformed variables to avoid seg fault on val access

425:

426: try {

427: } catch (const std::exception& e) {

428: stan::lang::rethrow\_located(e,current\_statement\_begin\_\_);

429: // Next line prevents compiler griping about no return

430: throw std::runtime\_error("\*\*\* IF YOU SEE THIS, PLEASE REPORT A BUG \*\*\*");

431: }

432:

433: // validate generated quantities

434:

435: // write generated quantities

436: }

437:

438: template <typename RNG>

439: void write\_array(RNG& base\_rng,

440: Eigen::Matrix<double,Eigen::Dynamic,1>& params\_r,

441: Eigen::Matrix<double,Eigen::Dynamic,1>& vars,

442: bool include\_tparams = true,

443: bool include\_gqs = true,

444: std::ostream\* pstream = 0) const {

445: std::vector<double> params\_r\_vec(params\_r.size());

446: for (int i = 0; i < params\_r.size(); ++i)

447: params\_r\_vec[i] = params\_r(i);

448: std::vector<double> vars\_vec;

449: std::vector<int> params\_i\_vec;

450: write\_array(base\_rng,params\_r\_vec,params\_i\_vec,vars\_vec,include\_tparams,include\_gqs,pstream);

451: vars.resize(vars\_vec.size());

452: for (int i = 0; i < vars.size(); ++i)

453: vars(i) = vars\_vec[i];

454: }

455:

456: static std::string model\_name() {

457: return "modele084e5e7a89\_2016\_06\_14\_tnd\_misclass";

458: }

459:

460:

461: void constrained\_param\_names(std::vector<std::string>& param\_names\_\_,

462: bool include\_tparams\_\_ = true,

463: bool include\_gqs\_\_ = true) const {

464: std::stringstream param\_name\_stream\_\_;

465: param\_name\_stream\_\_.str(std::string());

466: param\_name\_stream\_\_ << "pi0";

467: param\_names\_\_.push\_back(param\_name\_stream\_\_.str());

468: param\_name\_stream\_\_.str(std::string());

469: param\_name\_stream\_\_ << "pi1";

470: param\_names\_\_.push\_back(param\_name\_stream\_\_.str());

471: param\_name\_stream\_\_.str(std::string());

472: param\_name\_stream\_\_ << "VE";

473: param\_names\_\_.push\_back(param\_name\_stream\_\_.str());

474: param\_name\_stream\_\_.str(std::string());

475: param\_name\_stream\_\_ << "sens0";

476: param\_names\_\_.push\_back(param\_name\_stream\_\_.str());

477: param\_name\_stream\_\_.str(std::string());

478: param\_name\_stream\_\_ << "sens1";

479: param\_names\_\_.push\_back(param\_name\_stream\_\_.str());

480:

481: if (!include\_gqs\_\_ && !include\_tparams\_\_) return;

482: for (int k\_0\_\_ = 1; k\_0\_\_ <= N; ++k\_0\_\_) {

483: param\_name\_stream\_\_.str(std::string());

484: param\_name\_stream\_\_ << "p" << '.' << k\_0\_\_;

485: param\_names\_\_.push\_back(param\_name\_stream\_\_.str());

486: }

487:

488: if (!include\_gqs\_\_) return;

489: }

490:

491:

492: void unconstrained\_param\_names(std::vector<std::string>& param\_names\_\_,

493: bool include\_tparams\_\_ = true,

494: bool include\_gqs\_\_ = true) const {

495: std::stringstream param\_name\_stream\_\_;

496: param\_name\_stream\_\_.str(std::string());

497: param\_name\_stream\_\_ << "pi0";

498: param\_names\_\_.push\_back(param\_name\_stream\_\_.str());

499: param\_name\_stream\_\_.str(std::string());

500: param\_name\_stream\_\_ << "pi1";

501: param\_names\_\_.push\_back(param\_name\_stream\_\_.str());

502: param\_name\_stream\_\_.str(std::string());

503: param\_name\_stream\_\_ << "VE";

504: param\_names\_\_.push\_back(param\_name\_stream\_\_.str());

505: param\_name\_stream\_\_.str(std::string());

506: param\_name\_stream\_\_ << "sens0";

507: param\_names\_\_.push\_back(param\_name\_stream\_\_.str());

508: param\_name\_stream\_\_.str(std::string());

509: param\_name\_stream\_\_ << "sens1";

510: param\_names\_\_.push\_back(param\_name\_stream\_\_.str());

511:

512: if (!include\_gqs\_\_ && !include\_tparams\_\_) return;

513: for (int k\_0\_\_ = 1; k\_0\_\_ <= N; ++k\_0\_\_) {

514: param\_name\_stream\_\_.str(std::string());

515: param\_name\_stream\_\_ << "p" << '.' << k\_0\_\_;

516: param\_names\_\_.push\_back(param\_name\_stream\_\_.str());

517: }

518:

519: if (!include\_gqs\_\_) return;

520: }

521:

522: }; // model

523:

524: } // namespace

525:

526: typedef modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass stan\_model;

527:

528: #include <rstan/rstaninc.hpp>

529: /\*\*

530: \* Define Rcpp Module to expose stan\_fit's functions to R.

531: \*/

532: RCPP\_MODULE(stan\_fit4modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_mod){

533: Rcpp::class\_<rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass,

534: boost::random::ecuyer1988> >("stan\_fit4modele084e5e7a89\_2016\_06\_14\_tnd\_misclass")

535: // .constructor<Rcpp::List>()

536: .constructor<SEXP, SEXP>()

537: // .constructor<SEXP, SEXP>()

538: .method("call\_sampler",

539: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::call\_sampler)

540: .method("param\_names",

541: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::param\_names)

542: .method("param\_names\_oi",

543: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::param\_names\_oi)

544: .method("param\_fnames\_oi",

545: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::param\_fnames\_oi)

546: .method("param\_dims",

547: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::param\_dims)

548: .method("param\_dims\_oi",

549: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::param\_dims\_oi)

550: .method("update\_param\_oi",

551: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::update\_param\_oi)

552: .method("param\_oi\_tidx",

553: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::param\_oi\_tidx)

554: .method("grad\_log\_prob",

555: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::grad\_log\_prob)

556: .method("log\_prob",

557: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::log\_prob)

558: .method("unconstrain\_pars",

559: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::unconstrain\_pars)

560: .method("constrain\_pars",

561: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::constrain\_pars)

562: .method("num\_pars\_unconstrained",

563: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::num\_pars\_unconstrained)

564: .method("unconstrained\_param\_names",

565: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::unconstrained\_param\_names)

566: .method("constrained\_param\_names",

567: &rstan::stan\_fit<modele084e5e7a89\_2016\_06\_14\_tnd\_misclass\_namespace::modele084e5e7a89\_2016\_06\_14\_tnd\_misclass, boost::random::ecuyer1988>::constrained\_param\_names)

568: ;

569: }

570:

571: // declarations

572: extern "C" {

573: SEXP filee0843b44cb1( ) ;

574: }

575:

576: // definition

577:

578: SEXP filee0843b44cb1( ){

579: return Rcpp::wrap("2016\_06\_14\_tnd\_misclass");

580: }

581:

582:

Error in compileCode(f, code, language = language, verbose = verbose) :

Compilation ERROR, function(s)/method(s) not created! In file included from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/prim/mat.hpp:36:0,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/rev/mat.hpp:8,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math.hpp:4,

from C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/src/stan/model/model\_header.hpp:4,

from filee0843b44cb1.cpp:8:

C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/prim/mat/err/check\_positive\_ordered.hpp: In function 'bool stan::math::check\_positive\_ordered(const char\*, const char\*, const Eigen::Matrix<Scalar, -1, 1>&)':

C:/Users/huiying Chua/Documents/R/win-library/3.3/StanHeaders/include/stan/math/prim/mat/err/check\_positive\_ordered.hpp:39:67: warning: typedef 'size\_type' locally defined but no

Called from: compileCode(f, code, language = language, verbose = verbose)