**Supplement 2**

Simulation model

The simulation model was implemented in a as a discrete time model (time step:1 day) with given incidence functions for influenza and non-influenza ARI infections, as well as for the non-infectious control-eligible causes for hospitalization, possibly different for CP and non-CP subjects. Given an “all-or-none” vaccine model the vaccine effectiveness determined the number of subjects susceptible to influenza despite vaccination. For each iteration (day) cases and controls are generated in the following way.

1. The daily event probability on day for stratum was modeled as

where and are indicator variables for influenza susceptibility and CP status of stratum i. The number was drawn as pseudo-random number from a binomial distribution parameterized by the stratum size and the probability . The strata were indexed by influenza susceptibility, vaccination and CP status.

1. From the number of events, , the number of influenza and non-influenza infections was drawn as binomial pseudo-random number using the probability that an event was an ARI event
2. The number of influenza infections was drawn from using probability

which determined the number of non-influenza infections as . The number of non-ARI events follows from that as .

1. The number of events of a certain class then gave rise to study participants by drawing binomial pseudo-random numbers using event-specific -parameters (probability of hospitalization, given a particular event−see Table 1) and the vaccination effect on hospitalization probability given influenza infection ().
2. Given study participation, the number of vaccinated cases and controls of particular CP status was then stochastically chosen using the accuracy of CP status assessment ().
3. The number influenza susceptibles was then updated; susceptibility to other events remained universal.
4. Repeat 1 through 6 for the given duration of the influenza season.

After the end of the event generation step, if the numbers of cases and/or controls exceeded the target numbers, they were reduced by drawing a random binomial number of cases and controls, respectively, in each stratum. The binomial probabilities for cases and controls were calculated as the target numbers divided by the achieved numbers. For each of the resulting synthetic data sets, two logistic regression base models were formulated:

Model 1 (“crude”):

Model 2 (“adjusted”):

Note that represents the “crude” and represents the CP-adjusted coefficient. Parameter estimation was performed using unconditional logistic regression.

The simulation was implemented using R (version 3.1.1). The simulation model has twenty-four scalar parameters, four parameter vectors of dimension , one each for the influenza ARI, the non-influenza ARI and two for CP condition (non-CP and CP) daily incidence (Table 2). The model code is available as online supplement (Supplement 2).

Simulation code

This online supplement contains the R code used for simulation studies. This online supplement contains four R files:

1. Hosp TND GUI.R
2. Hosp TND simulation.R
3. parDefaultSimple.Rdata
4. Hosp TND simulation run.R

These files should be copied into one directory. The first file (Hosp TND GUI.R ) represents the construction of a simple graphical user interface (GUI) for parameter input. For this to run the following R packages are required: “gWidgets2”, “gWidgets2RGtk2” and “tcltk2”. The file “parDefaultSimple.Rdata” contains the default parameter values. Opening the file “Hosp TND simulation run.R” the working directory has to be given (use forward slashes for the path). Running the code contained in that file should then invoke the GUI that allows to modify parameter values. If you first get an error message such as “libatk-1.0-0.dll is missing”, first try to close the error window and check “ok” if prompted to install GTK. Otherwise, check [here](https://stat.ethz.ch/pipermail/r-help/2007-February/126545.html) for possible solutions to the problem.

Choose to “select a file?” → “Yes” and choose the file “Hosp TND simulation.R” when prompted. The simulation can then be re-run by only running the command “pardef()” which will again bring up the GUI.