Example of use of “StateDiagrams.m”

Initialization

Loads the StateDiagrams.m package (change the directory to the folder you are using)

In[1]:=
```
Directory[];
```

In[2]:=
```
SetDirectory[NotebookDirectory[]];
<< "StateDiagrams_mma-v12.m"
```

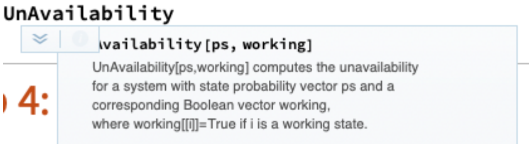
A list of the available functions in the package is obtained by the command:

In[4]:=
```
Names["Dependability`StateDiagrams`*"]
```

Out[4]=
```
{ArrangeMatrix, DownTimeDensity, DownTimeDensityLaplace,
 GenHomogeneousSystem, KoutofN, KoutofNMode, KroenSum, LabelList, MDT,
 MergeStates, MTBF, MTFF, MTTF, MUT, ParallelMode, PlotDiagram, ProbStationary,
 ProbTransient, RelFunc, SeriesMode, SetDiagonal, StepPlot, UnAvailability}
```

Check syntax in the examples in task 1 below, and by ?UnAvailability, etc.. of mouseover.

```
UnAvailability
```



In[5]:=
```
? SeriesMode;
```

Out[5]=
```
Missing[UnknownSymbol, SeriesMode;]
```

Lab 4: Dependability modelling

Students: August Heltne-Karlson and Erik Turøy Midtun

System description

In this part you are going to develop an analytic model with the objective to study the dependability with respect to airport availability and time until reopening of a runway after a snowfall.

In the modelling of the airport in this assignment you need reliability block diagrams and Markov models [see Chapter 7 in the textbook].

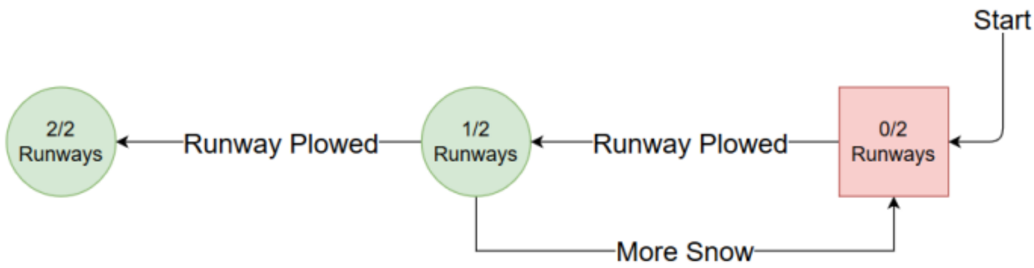
Task 1: Model of two plowing trucks [warm-up, 0%]

Task 2: Probability of open runways after snow fall [25%]

You are going to make a model to obtain the transient probability $q_i(t)$ that i runways are open at time t.
Assume that
- you have two runways, and that no trucks fail during plowing,
- the expected snowing time is $1/\kappa$ and plowing time is $1/\gamma$,
- when both runways are cleaned and reopened, it is assumed that snowing stops
(this means that we are only interested in *the time until first reopening* after the snow fall closed the airport), and
- two trucks can not plow the same runways simultaneously.

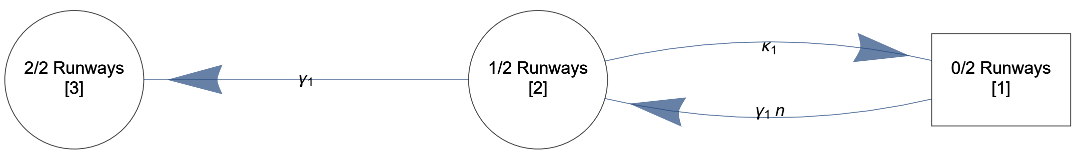
The objective is to study the transient period from *closing both runways at t=0,* and until the airport is partly (1 runway), and fully reopened (both runways).

2.1. Define the system state variable(s) and corresponding events and make a Markov model to obtain the transient state probabilities for *n* trucks.
The system state variables are the number of available runways. The corresponding events are plowing the runways and snowing filling up the runways.



In[50]:=
```
Q = Table[",", {3}, {3}];
Q[[1, 2]] = n * γ1;
Q[[1, 3]] = 0;
Q[[2, 1]] = κ1;
Q[[2, 3]] = γ1;
Q[[3, 1]] = 0;
Q[[3, 2]] = 0;
M2 = SetDiagonal[Transpose[Q]];
L2 = {
    "0/2 Runways\n[1]",
    "1/2 Runways\n[2]",
    "2/2 Runways\n[3]"};
W2 = {False, True, True};
PlotDiagram[M2, W2, L2]
```

Out[60]=



2.2. Use the package provided to obtain the probability $q_i(t)$ that *i* runways are reopened at time t, given that they were closed at t=0 after a heavy snow fall (general for *n* trucks).

In[190]:=
```
P2[t_] := ProbTransient[M2, {1, 0, 0}, x] /. {x → t}
P2[t] // Simplify;
For[i = 0, i < 3, i++, Print["q", i , "=", Take[P2[t], {i + 1}] // Simplify, "\n\n"]]
```

$$
\begin{aligned}
q0=\Bigg\{ & \frac{1}{2\left((-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2\right)} e^{-\frac{1}{2}t\left(\gamma_1+n\gamma_1+\kappa_1+\sqrt{-4n\gamma_1^2+((1+n)\gamma_1+\kappa_1)^2}\right)} \Bigg(\left(1+e^{t\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}}\right)(-1+n)^2\gamma_1^2+ \\
& \kappa_1\Bigg(\left(1+e^{t\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}}\right)\kappa_1+\left(-1+e^{t\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}}\right) \\
& \sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}\Bigg)+\gamma_1\Bigg(2\left(1+e^{t\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}}\right)(1+n)\kappa_1- \\
& \left(-1+e^{t\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}}\right)(-1+n)\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}\Bigg)\Bigg)\Bigg\}
\end{aligned}
$$

$$
q1=\left\{\frac{e^{-\frac{1}{2}t\left(\gamma_1+n\gamma_1+\kappa_1+\sqrt{-4n\gamma_1^2+((1+n)\gamma_1+\kappa_1)^2}\right)}\left(-1+e^{t\sqrt{-4n\gamma_1^2+((1+n)\gamma_1+\kappa_1)^2}}\right)n\gamma_1}{\sqrt{-4n\gamma_1^2+((1+n)\gamma_1+\kappa_1)^2}}\right\}
$$

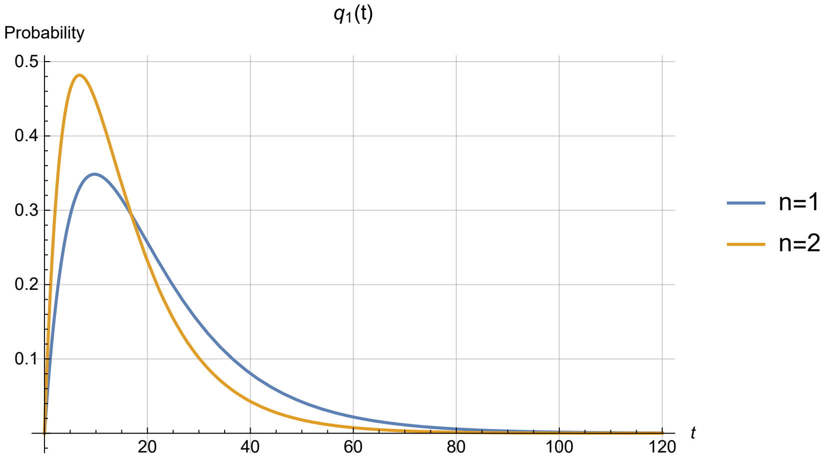
$$
\begin{aligned}
q2=\Bigg\{ & \frac{1}{2\left(-4n\gamma_1^2+((1+n)\gamma_1+\kappa_1)^2\right)} e^{-\frac{1}{2}t\left(\gamma_1+n\gamma_1+\kappa_1+\sqrt{-4n\gamma_1^2+((1+n)\gamma_1+\kappa_1)^2}\right)} \\
& \Bigg(-\left(1+e^{t\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}}-2e^{\frac{1}{2}t\left((1+n)\gamma_1+\kappa_1+\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}\right)}\right)(-1+n)^2\gamma_1^2- \\
& \kappa_1\Bigg(\left(1+e^{t\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}}-2e^{\frac{1}{2}t\left((1+n)\gamma_1+\kappa_1+\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}\right)}\right)\kappa_1+ \\
& \left(-1+e^{t\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}}\right)\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}\Bigg)- \\
& (1+n)\gamma_1\Bigg(2\left(1+e^{t\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}}-2e^{\frac{1}{2}t\left((1+n)\gamma_1+\kappa_1+\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}\right)}\right)\kappa_1+ \\
& \left(-1+e^{t\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}}\right)\sqrt{(-1+n)^2\gamma_1^2+2(1+n)\gamma_1\kappa_1+\kappa_1^2}\Bigg)\Bigg)\Bigg\}
\end{aligned}
$$

2.3. Plot $q_1(t)$ and $q_2(t)$ for *n=1* and *n=2* and compare the results. Use {$\gamma \to 0.1, \kappa \to 1/60$}[1/minutes]

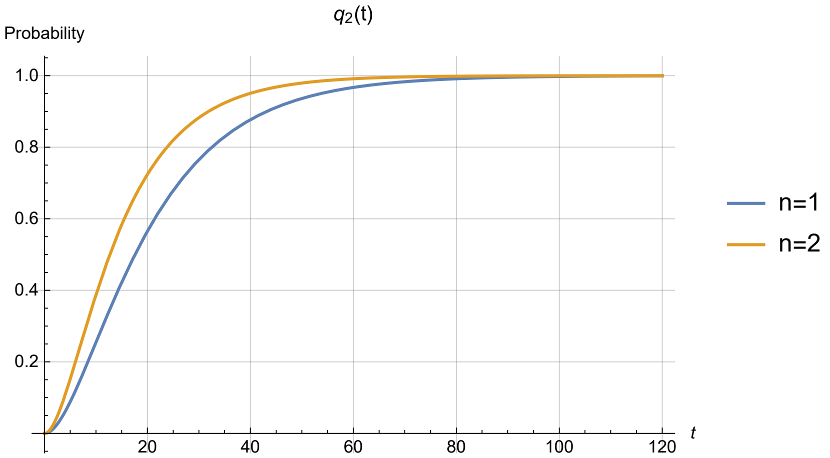
Plot all state probabilities for the first two hours (120 minutes) and discuss what you observe. Use {$\gamma \to 0.1, \kappa \to 1/60$} [1/minutes]

In[64]:=
```
𝒫11 = {γ1 → 1/10, κ1 → 1/60, n→1};
q0n1[t_] = Take[P2[t], {1}] /. 𝒫11 // Simplify;
q1n1[t_] = Take[P2[t], {2}] /. 𝒫11 // Simplify;
q2n1[t_] = Take[P2[t], {3}] /. 𝒫11 // Simplify;

𝒫12 = {γ1 → 1/10, κ1 → 1/60, n→2};
q0n2[t_] = Take[P2[t], {1}] /. 𝒫12 // Simplify;
q1n2[t_] = Take[P2[t], {2}] /. 𝒫12 // Simplify;
q2n2[t_] = Take[P2[t], {3}] /. 𝒫12 // Simplify;

Plot[{
   Legended[q1n1[t], "n=1"],
   Legended[q1n2[t], "n=2"]},
  {t, 0, 120}, PlotRange → All, PlotLabel → "q1(t)",
  AxesLabel → {Automatic, "Probability"}, GridLines → Automatic]

Plot[{
   Legended[q2n1[t], "n=1"],
   Legended[q2n2[t], "n=2"]},
  {t, 0, 120}, PlotRange → All, PlotLabel → "q2(t)",
  AxesLabel → {Automatic, "Probability"}, GridLines → Automatic]
```

Out[72]=



Out[73]=



With (n=2) two plow trucks we see that in the beginning the probability of being in state 1, the state with no runways available, is lower than in the case with only one truck. We also see that with two trucks we get to the steady state(100% probability for both runways available) faster than with one truck. The overall availability is thus higher with two trucks.

Furthermore, find the maximum reopening time T_{max} for which the probability is ≥0.95,
P(T<T_{max})≥0.95.

Compare n=1 and n=2.

In[74]:=
```
root_n1 = FindRoot[(q2n1[t_1]) - 0.95 == 0, {t_1, 0.1}] ;
root_n2 = FindRoot[(q2n2[t_2]) - 0.95 == 0, {t_2, 0.1}];
Print["n=1: ", root_n2]
Print["n=2: ", root_n2 ]
Print["Difference t_1 and t_2: ", t_1 - t_2 /. root_n1 /. root_n2 ,
 "min (", ((t_1 - t_2) / t_1 /. root_n1 /. root_n2 ) * 100, "%)"]
```

```
n=1: {t_2 → 39.8451}

n=2: {t_2 → 39.8451}

Difference t_1 and t_2: 13.8314min (25.7681%)
```

The maximum time needed to fully reopen (2/2 runways) with at least 95% probability is a lot faster when having two trucks instead of one. We can see that two trucks reopens at approximately 14 minutes(~26%) faster than when only having one truck.

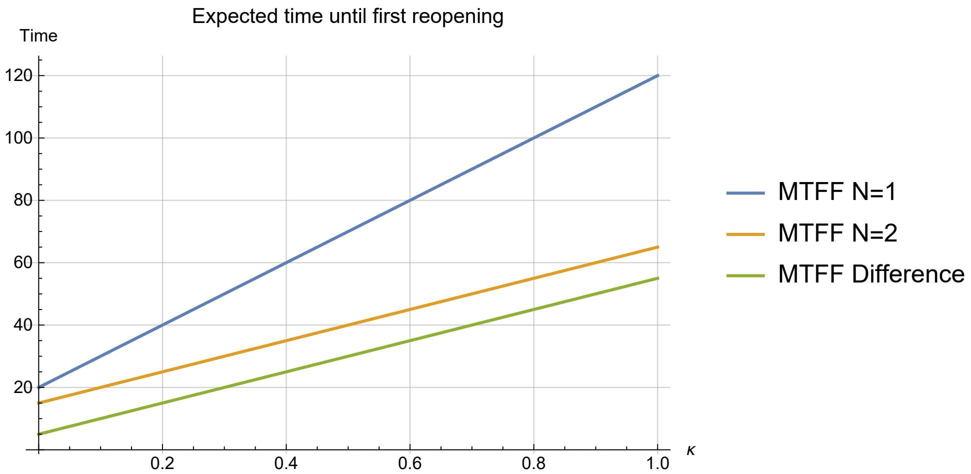
2.4. Compare *the expected time until first reopening* with n=1 and n=2.

Plot the *relative difference in the expected time until first reopening* (gain) for n=1 and n=2 as the expected snowing time 1/*κ* changes.

We can use MTFF if we change the model to be {True, True, False} instead of {False, True, True}.

In[79]:=
```
𝒫n1= {n→1};
𝒫n2= {n→2};
𝒫24 = {γ1 → 1/10};
𝒲24 = {True, True, False};
mtffN1 = MTFF[ℳ2, 𝒲24] /. 𝒫n1 /. 𝒫24 ;
mtffN2 = MTFF[ℳ2, 𝒲24] /. 𝒫n2 /. 𝒫24 ;
mtffN21[k_] := mtffN1 /. {κ1 → k}
mtffN22[k_] := mtffN2 /. {κ1 → k}
diff [k_] := mtffN1 - mtffN2 /. {κ1 → k}

Plot[{
  Legended[mtffN21[κ], "MTFF N=1"],
  Legended[mtffN22[κ], "MTFF N=2"],
  Legended[diff[κ], "MTFF Difference"]},
 {κ, 0, 1}, PlotRange → All, PlotLabel → "Expected time until first reopening",
 AxesLabel → {Automatic, "Time"}, GridLines → Automatic]
```

Out[88]=



Discuss the gain you get by adding one truck (n=1 -> n=2) when κ changes. Use {γ→0.1} [1/minutes]

The smallest difference between MTFF for N1 and MTFF for N2 is 33% at k = 0. As the entire airport will stop if the runways are closed, having the ability to reopen as quickly as possible is very important. At κ = 1/60 as used earlier in this task, the difference is approximately 5.8 minutes. If κ = 1/60 means that the runways need to be plowed at average once per hour, this ends up costing the airport almost 70 minutes if it stays open from 08-20. This is most likely a too large time loss for the airport to operate properly and as such we suggest having two trucks as long the it is economically viable.

What is the minimum and maximum gain (and for which κ value)?

In[89]:=
```
Print["MTFF for N=1: ", mtffN1 // Simplify]
Print["MTFF for N=2: ", mtffN2 // Simplify]
Print["Difference: ", mtffN1 - mtffN2 // Simplify]
```

MTFF for N=1: $20\,(1 + 5\,\kappa_1)$

MTFF for N=2: $15 + 50\,\kappa_1$

Difference: $5 + 50\,\kappa_1$

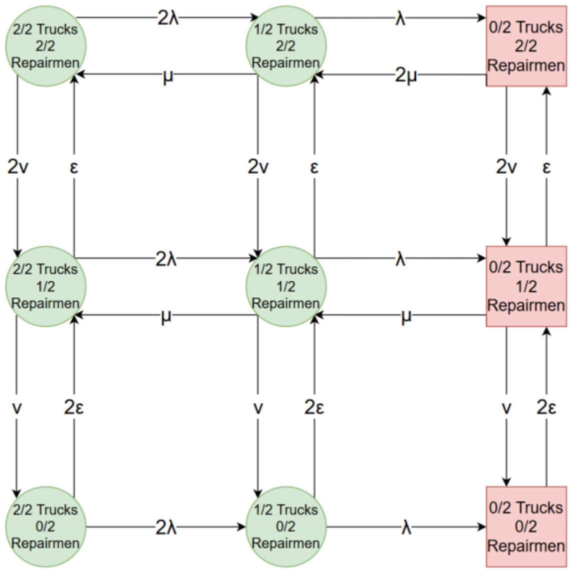
As we can see, the gain is linear. This leads to the minimum gain being where $\kappa = 0$ -> diff = 5 and the maximum being where $\kappa = 1$ -> 55.

Task 3: Two plowing trucks with two repairmen who might go on sick-leave [30%]

The repair of the trucks depends on a repairman.
In the model from task 1 we assumed that we have one repairman per truck.
However, the repairmen might get sick (intensity ν) and are recovered (intensity ϵ).
Make the necessary changes in the model from task 1 to take into account that the repair of a truck might get delayed due to (a) sick repairman(men).

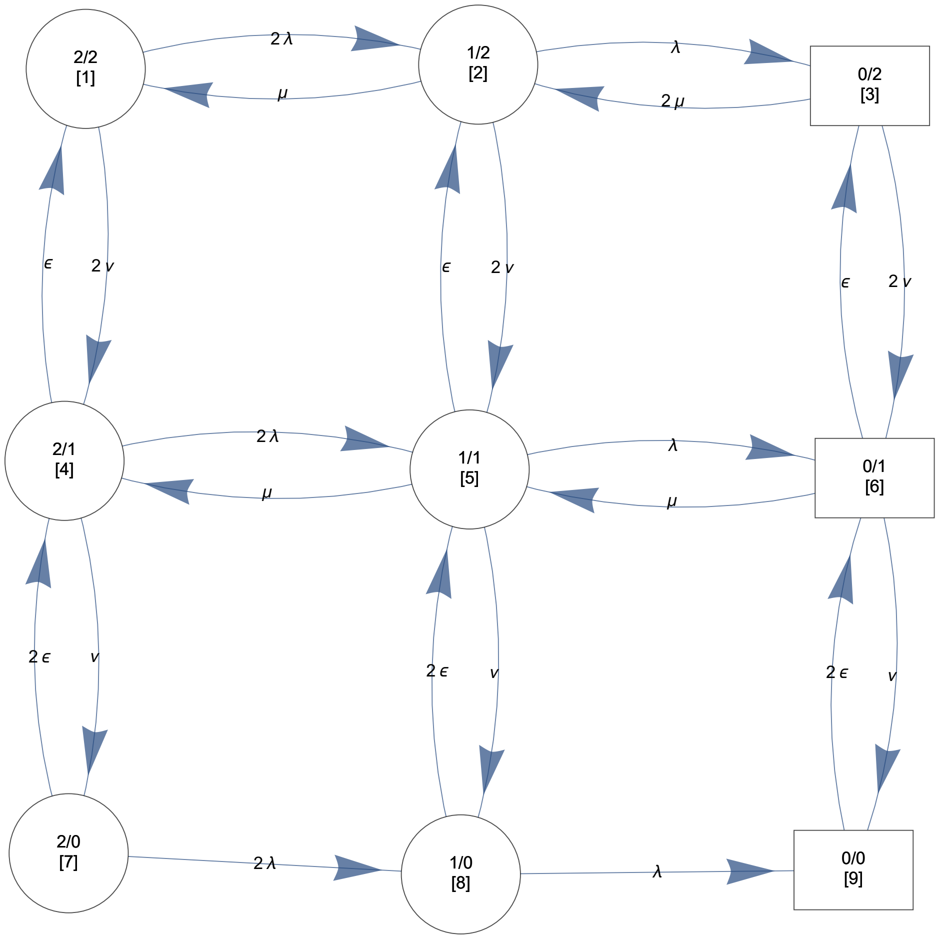
3.1. Define the system state variable(s) and corresponding events, and make a Markov model to obtain state probabilities.

The system state variables are the number of trucks and repairmen available. The corresponding events are trucks failing and getting fixed and the repairmen getting sick and getting better.



In[92]:=
```
Q3 = {
    {, 2λ, 0, 2ν, 0,   0,   0, 0,  0},
    {μ, ,  λ, 0,   2ν, 0,   0, 0,  0},
    {0, 2μ, , 0,   0,   2ν, 0, 0,  0},
    {ϵ, 0,  0, ,   2λ,  0,   ν, 0,  0},
    {0, ϵ,  0, μ,   ,   λ,   0, ν,  0},
    {0, 0,  ϵ, 0,  μ,    ,   0, 0,  ν},
    {0, 0,  0, 2ϵ, 0,   0,    , 2λ, 0},
    {0, 0,  0, 0,   2ϵ, 0,   0, ,  λ},
    {0, 0, 0,  0,   0,   2ϵ, 0, 0,  }
  };
M3 = SetDiagonal[Transpose[Q3]];
L3 = {"2/2\n[1]", "1/2\n[2]", "0/2\n[3]",
    "2/1\n[4]", "1/1\n[5]", "0/1\n[6]",
    "2/0\n[7]", "1/0\n[8]", "0/0\n[9]"};
W3 = {True, True, False, True, True, False, True, True, False};
PlotDiagram[M3, W3, L3]
```

Out[96]=



3.2. Use the package provided to obtain the probability that n=0, 1, and 2 trucks are available for $\mathcal{P}_2$ = {$\lambda_2 \to$ 0.1, $\mu_2 \to$ 1, $\nu \to$ 0.01, $\epsilon \to$ 0.5}[1/days]

In[97]:=
```
𝒫32 = {λ→0.1,  μ → 1,  ν → 0.01, ϵ → 0.5};
P3[t_] := ProbTransient[(ℳ3 /. 𝒫32), {1, 0, 0, 0, 0, 0, 0, 0, 0}, x] /. {x → t}

PS31 = ProbStationary[(ℳ3 /. 𝒫32)] ;
n2 = Take[PS31, {1}] + Take[PS31, {4}] + Take[PS31, {7}];
n2t[t_] = Take[P3[t], {1}] + Take[P3[t], {4}] + Take[P3[t], {7}] // Simplify;
n1 = Take[PS31, {2}] + Take[PS31, {5}] + Take[PS31, {8}];
n1t[t_] = Take[P3[t], {2}] + Take[P3[t], {5}] + Take[P3[t], {8}] // Simplify;
n0 = Take[PS31, {3}] + Take[PS31, {6}] + Take[PS31, {9}];
n0t[t_] = Take[P3[t], {3}] + Take[P3[t], {6}] + Take[P3[t], {9}] // Simplify;
Print["n=2\n PS:", n2, "\n P(t):", n2t[t]]
Print["n=1\n PS:", n1, "\n P(t):", n1t[t]]
Print["n=0\n PS:", n0, "\n P(t):", n0t[t]]
```

n=2
PS:{0.826121}
P(t): $\{0.826121 + 0.00708125\, e^{-2.25089\, t} - 0.000326728\, e^{-1.97019\, t} + 0.0120637\, e^{-1.37775\, t} - 0.0133304\, e^{-1.2145\, t} + 0.119444\, e^{-1.11537\, t} + 0.15563\, e^{-1.0313\, t} - 0.107927\, e^{-1.02\, t} + 0.00124419\, e^{-0.51\, t}\}$

n=1
PS:{0.165329}
P(t): $\{0.165329 - 0.0145381\, e^{-2.25089\, t} + 0.00057375\, e^{-1.97019\, t} - 0.014617\, e^{-1.37775\, t} + 0.0148217\, e^{-1.2145\, t} - 0.110669\, e^{-1.11537\, t} - 0.127615\, e^{-1.0313\, t} + 0.0873579\, e^{-1.02\, t} - 0.000643598\, e^{-0.51\, t}\}$

n=0
PS:{0.0085491}
P(t): $\{0.0085491 + 0.00745682\, e^{-2.25089\, t} - 0.000247021\, e^{-1.97019\, t} + 0.00255327\, e^{-1.37775\, t} - 0.00149131\, e^{-1.2145\, t} - 0.00877489\, e^{-1.11537\, t} - 0.0280144\, e^{-1.0313\, t} + 0.020569\, e^{-1.02\, t} - 0.000600595\, e^{-0.51\, t}\}$

3.3. Compare the two models, without (task 1) and with (task 2) repairmen on sick leave with respect to the unavailability and the probability that one truck has failed.
Change the repair intensity β (assume it partly depends on the skills) and discuss whether it is sufficient to have one truck or if you need two.
You first need to define what you think is the maximum acceptable probability of not having a truck available.

In[109]:=
```
𝒫33 = {λ→0.1, ν → 0.01, ϵ → 0.5};
PS33 = ProbStationary[(ℳ3 /. 𝒫33)] ;
n233[β_] := (Take[PS33, {1}] + Take[PS33, {4}] + Take[PS33, {7}]) /. {μ → β} ;
n133 [β_] := (Take[PS33, {2}] + Take[PS33, {5}] + Take[PS33, {8}]) /. {μ → β};
n033[β_] := (Take[PS33, {3}] + Take[PS33, {6}] + Take[PS33, {9}]) /. {μ → β};

n21 [β_] := (Take[PS1, {1}]) /. {μ1 → β, λ1 → 0.1}
n11 [β_] := (Take[PS1, {2}]) /. {μ1 → β, λ1 → 0.1}
n01[β_] := (Take[PS1, {3}]) /. {μ1 → β, λ1 → 0.1}

Print["B=0.5"]
Grid[{{"Task", "N0", "N1", "N2"},
   {1, n01 [0.5], n11 [0.5] , n21 [0.5]}, {3, n033 [0.5], n133 [0.5] , n233 [0.5]}},
  Background → {None, {GrayLevel[0.7`], {White}}}, Dividers → {Black, {2 → Black}},
  Frame → True, Spacings → {2, {2, {0.7`}, 2}}]
Print["B=1"]
Grid[{{"Task", "N0", "N1", "N2"}, {1, n01 [1], n11 [1] , n21 [1]},
   {3, n033 [1], n133 [1] , n233 [1]}}, Background → {None, {GrayLevel[0.7`], {White}}},
  Dividers → {Black, {2 → Black}}, Frame → True, Spacings → {2, {2, {0.7`}, 2}}]
```

B=0.5

Out[118]=

Task	N0	N1	N2
1	{0.0277778}	{0.277778}	{0.694444}
3	{0.0285189}	{0.277662}	{0.693819}

B=1

Out[120]=

Task	N0	N1	N2
1	{0.00826446}	{0.165289}	{0.826446}
3	{0.00852009}	{0.165334}	{0.826146}

Assuming that the repair intensity 1/day means that it will on average take a day to repair a truck, the airport needs to have two trucks. Having all trucks stop working at the same time means that it will be around a day before you can clear the runway. This would bring a complete halt to all air traffic in the area and is catastrophic for the airport. Therefore having an additional truck to lower the probability that they all fail at the same time and decreasing the time between entering the failure state and leaving it is very important for the airport to function over long periods.

One should aim for at least 99% probability of having at least one truck available when it snows simply because of how much of a problem missing a truck when you need it is. However, this is more achievable than our model suggests. The plowing trucks are only required when it is snowing and one can use this to perform maintenance on them when they’re not needed in order to reduce the probability of them not working when it is snowing

3.4. Obtain the mean time to first failure, and mean time to failure. Are they the same or different? Why?

In[193]:=
```
Print["MTFF:", MTFF[(ℳ3 /. 𝒫32), 𝒲3]]
Print["MTTF:", MTTF[(ℳ3 /. 𝒫32), 𝒲3]]
```

```
MTFF:64.9673
MTTF:64.1323
```

They are different. MTFF is the expected time between node 1(2 trucks and 2 repairmen available) to one of the states with 0 trucks available at t=0. MTTF is the mean time to failure from an arbitrary time when the system is in stationary state and working. As node 2/2 is the most resilient node in the network, MTFF is slightly longer than MTTF as MTTF can start in a less resilient node. The different between them is rather small as node 2/2 is the most probable state to be in

Task 4: Performability [15%] [no assistance]

The focus in this task is on what is called *Performability* - "a measurement of how a degradable system performs" [1],[2][3].
In a performability model you combine the dependability model (here: truck availability model from task 3) with the performance (here: probability of reopening runways from task 2).
Let's set the guaranteed maximum time to reopen to T_{max} = 45 minutes.

4.1. What is the probability $P(T < T_{max})$ that it takes less than T_{max} = 45 minutes to reopen both runways?
Calculate it for n=0, 1 and 2 trucks and using {γ→ 0.1,κ→ 1/60} [1/minutes] from task 2.

```
In[123]:= 𝒫41 = {γ1 → 1/10, κ1 → 1/60, n→0};
q2n0[t_] = Take[P2[t], {3}] /. 𝒫41 // Simplify;

N0Tmax = q2n0[45];
N1Tmax = Round[q2n1[45], 0.00000001];
N2Tmax = Round[q2n2[45], 0.0000001];
Print["For N=0: ", N0Tmax]
Print["For N=1: ", N1Tmax]
Print["For N=2: ", N2Tmax]

For N=0: {0}
For N=1: {0.91132}
For N=2: {0.96807}
```

4.2. Combine the probabilities (which are conditioned on n=0,1, and 2) with the model for truck availability from task 3 to obtain the *overall probability* that it takes less than T_{max} = 45 minutes to reopen both runways.
Use 𝒫2 = {λ2 → 0.1, μ2 → 1, ν → 0.01, ϵ → 0.5} in the truck availability model.

```
In[131]:= T45max = n0t[45] * N0Tmax + n1t[45] * N1Tmax + n2t[45] * N2Tmax
Out[131]= {0.950412}
```

4.3. Which factors contributes to the reopening time?
Discuss what you would have done if you were asked to reduce the guaranteed maximum time to reopen to T_{max} = 30 minutes, without changing the guaranteed probability $P(T < T_{max})$.
We cannot do anything with the snowing time κ and we have seen that the effects of repairmen getting sick and better is negligible. So the only factors we can change is the plowing time γ, break down intensity λ and the repairing intensity μ.

Calculate the new probabilities first, identify the factors that contributes, and do a qualitative discussions about (counter)measures that could be taken to ensure that the guaranteed probability $P(T < T_{\max})$ is provided.

In[132]:=
```
P₂[t_] = ProbTransient[M₂, {1, 0, 0}, x] /. {x → t};
P43 = {λ→0.00001, μ→1, ν → 0.01, ϵ → 0.5};
P43[t_] = ProbTransient[(M₃ /. P43), {1, 0, 0, 0, 0, 0, 0, 0, 0}, x] /. {x → t};
n2t43[t_] = Take[P43[t], {1}] + Take[P43[t], {4}] + Take[P43[t], {7}] // Simplify;
n1t43[t_] = Take[P43[t], {2}] + Take[P43[t], {5}] + Take[P43[t], {8}] // Simplify;

P43n1 = {κ₁ → 1/60, n→1};
q2n143[t_] = Take[P₂[t], {3}] /. P43n1 // Simplify;

P₂[t_] := ProbTransient[M₂, {1, 0, 0}, x] /. {x → t}
P43n2 = {κ₁ → 1/60, n→2};
q2n243[t_] = Take[P₂[t], {3}] /. P43n2 // Simplify;
T30λSmax = n1t43[30] * q2n143[30] + n2t43[30] * q2n243[30];
T30λOmax = n1t[30] * q2n143[30] + n2t[30] * q2n243[30];
valueλS = FindRoot[(T30λSmax) - (1 - First[1 - T45max]) == 0, {γ₁, 0.15}];
valueλO = FindRoot[(T30λOmax) - (1 - First[1 - T45max]) == 0, {γ₁, 0.15}];
Print["Original γ₁ value: 0.1"]
Print["Searching for γ₁ that gives P(T<30)=~0.95 with all other original values:",
 valueλO]
Print["Ignoring truck failures (λ→0.00001) gives:", valueλS]
```

```
Original γ₁ value: 0.1

Searching for γ₁ that gives P(T<30)=~0.95 with all other original values:{γ₁ → 0.145861}

Ignoring truck failures (λ→0.00001) gives:{γ₁ → 0.130729}
```

After running calculations with λ -> 0, we can see that even while constantly having 2 plow trucks available, we still need to increase γ with 30%. This is because the plowing time is the bottleneck. As the plowing time is dependent on two variables: plowing intensity and snowing intensity, we have to change them in order to deliver P(T < Tmax). However, snowing intensity is out of the airport’s control so we need to increase the plowing intensity. Two examples of measures to increase plowing intensity are stronger engines to speed up the trucks and allowing multiple trucks to work on the same runway. If one combines this with maintenance routines to reduce the probability that trucks fail and training repairmen so they can repair trucks faster one might be able to hit P(t < 30) >= 95%.

Task 5: Structural model of the whole airport [30%] [no assistance]

The system is “up/working” when

- both runways, AND
- at least 6 of 10 gates, AND
- one of two deicing stations

are available.

5.1. What do you have to assume to use a Reliability Block Diagram (RBD)? What does this mean in the context of the airport system? Is this realistic?

The following assumptions about RBDs are listed in the book:

1. If the system is made up by more than one subsystem, each system fails independently of all other subsystems, i.e. independently of the state of the system.

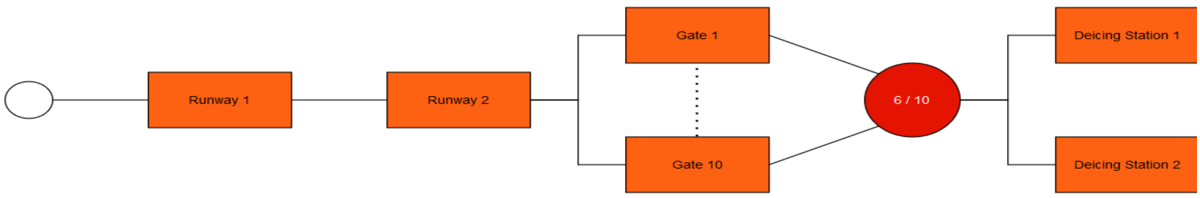
2. If a subsystem has failed, its service is restored independently of the state of all other subsystems. (This assumption requires for instance a dedicated repairman and test equipment for each subsystem.)

3. The system behaves as intended. For instance, a fault-tolerant system handles faults as long as there are remaining (spare) resources, service restoration actions are successful, and failures (errors) do not propagate from one subsystem to others.

Note that if any of these assumptions deviates from the true behavior of the system, the analysis loses its precision.

For the system in question, this means that we have to assume that the runways are completely independent from the gates and the deicing stations and vice versa. These components are generally independent, but events that can cause a failure for one component can most likely also hurt the other components. An example for this is a snowstorm. As we are modelling the airport when it is snowing it is not completely realistic that all the component are failing independently.

5.2. Make an Reliability Block Diagram (RBD) of the total airport system



5.3. Make reliability functions for

- R(t,λ) (single component),

```
In[149]:= R[t_, λ_] = Exp[-λ * t];
```

- R_s(t,n,λ), for an n-serial structure

```
In[150]:= Rs[t_, n_, λ_] = Product[R[t, λ], {i, 1, n}];
```

- R_p(t,n,λ), for an n-parallel structure

```
In[151]:= Rp[t_, n_, λ_] = 1 - Product[1 - R[t, λ], {i, 1, n}];
```

- R_{kofn}(t,k,n,λ), for a k-of-n structure

```
In[152]:= Rkofn[t_, k_, n_, λ_] = Sum_{i=k}^{n} Binomial[n, i] * R[t, λ]^i * (1 - R[t, λ])^(n-i);
```

Check function by `Integrate[Rkofn[t, 6, 10, λ], {t, 0, ∞}, Assumptions → λ > 0]` = $\frac{1627}{2520\,\lambda}$. Which property have you now calculated?

In[153]:=
```
Integrate[Rkofn[t, 6, 10, λ], {t, 0, ∞}, Assumptions → λ > 0]
```

Out[153]= $\frac{1627}{2520\,\lambda}$

We have calculated the mean time to first failure (MTFF) of the K-of-N system

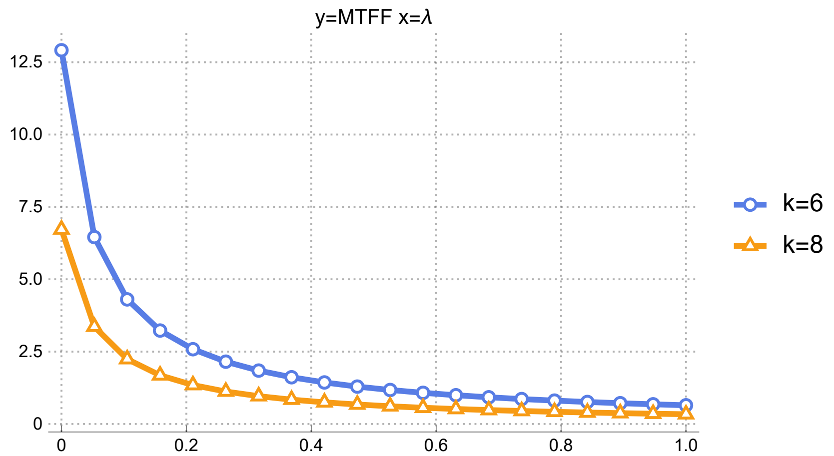
Explain how $R_s(t,n,\lambda)$ and $R_p(t,n,\lambda)$ can be replaced by $R_{kofn}(t,k,n,\lambda)$.

$\mathbf{R_s}$ is essentially the same as $\mathbf{R_{kofn}}$ with k=n and $\mathbf{R_p}$ is $\mathbf{R_{kofn}}$ with k=1. Later when we add more runways than 2, we change the runway reliability function to be $R_{kofn}(t,2,n,\lambda)$.

Plot the integral (from 0 to ∞) of Subscript[R, kofn](t,6,10,λ) and Subscript[R, kofn](t,8,10,λ) for 20 values of λ in the range [0,1] (hint:use Table and ListLinePlot)

In[154]:=
```
k6₅₃ = Table[
   Integrate[Rkofn[t, 6, 10, λ], {t, 0, ∞}, Assumptions → λ > 0], {λ, 1/20, 1, 1/20}];
k8₅₃ = Table[Integrate[Rkofn[t, 8, 10, λ], {t, 0, ∞}, Assumptions → λ > 0],
   {λ, 1/20, 1, 1/20}];
ListLinePlot[{k6₅₃, k8₅₃}, PlotRange → All, PlotLabel → "y=MTFF x=λ",
 AxesLabel → {λ, "MTFF"}, GridLines → Automatic, DataRange → {0, 1},
 PlotLegends → {"k=6", "k=8"}, PlotTheme → "Business"]
```

Out[156]=



5.4. Use your reliability functions to determine the $R_{tot}(t)$ for the airport, compliant with your RBD.

In[157]:=
```
Rtot[t_] = (Rs[t, 2, λ1] * Rkofn[t, 6, 10, λ2] * Rp[t, 2, λ3])
```

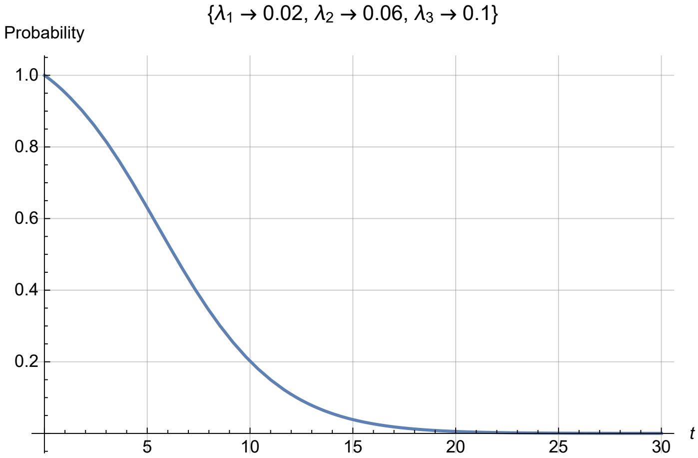
Out[157]= $$\frac{e^{-2t\lambda_1-6t\lambda_2}\left(1-e^{-t\lambda_2}\right)^4\left(126-560\,e^{t\lambda_2}+945\,e^{2t\lambda_2}-720\,e^{3t\lambda_2}+210\,e^{4t\lambda_2}\right)\left(1-\left(1-e^{-t\lambda_3}\right)^2\right)}{\left(-1+e^{t\lambda_2}\right)^4}$$

5.5. Plot $R_{tot}(t)$ for $\mathcal{P}_\lambda$ = {λ1 → 0.02, λ2 → 0.06, λ3 → 0.1}[1/days]

What is the probability that the airport runs without interruptions for 1 day, 1 week, and 1 month (including night shifts)?

In[158]:=
```
𝒫λ = {λ1 → 0.02, λ2 → 0.06, λ3 → 0.1};
Plot[ {Rtot[t] /. 𝒫λ}, {t, 0, 30}, PlotRange → All, PlotLabel → 𝒫λ,
 AxesLabel → {Automatic, "Probability"}, GridLines → Automatic]
Grid[{{"Days", "Rtot(%) 𝒫λ"}, {1, (Rtot[1] /. 𝒫λ) * 100}, {7, (Rtot[7] /. 𝒫λ) * 100},
  {30, (Rtot[30] /. 𝒫λ) * 100}}, Background → {None, {GrayLevel[0.7`], {White}}},
 Dividers → {Black, {2 → Black}}, Frame → True, Spacings → {2, {2, {0.7`}, 2}}]
```

Out[159]=



Out[160]=

Days	R_{tot} (%) $\mathcal{P}_\lambda$
1	95.1963
7	43.2636
30	0.00682305

From the table above, we can see the probabilities that the airport runs without interruption for 1 day, 1week and 1 month.

5.6. What is the main cause of failure? Which change(s) will have the highest effect?

In[161]:=
```
Grid[{{"Days", "Runways", "Gates", "Deicing" },
  {1, Rs[1, 2, 0.02], Rkofn[1, 6, 10, 0.06], Rp[1, 2, 0.1]},
   {7, Rs[7, 2, 0.02], Rkofn[7, 6, 10, 0.06], Rp[7, 2, 0.1]},
  {30, Rs[30, 2, 0.02], Rkofn[30, 6, 10, 0.06], Rp[30, 2, 0.1]
  }}, Background → {None, {GrayLevel[0.7`], {White}}},
 Dividers → {Black, {2 → Black}}, Frame → True, Spacings → {2, {2, {0.7`}, 2}}]
```

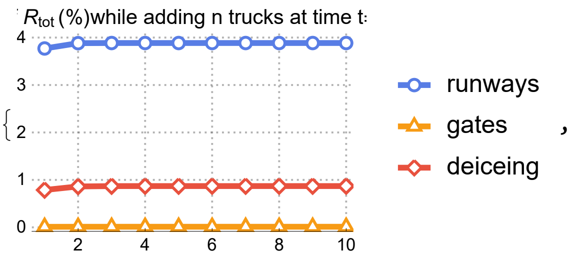
Out[161]=

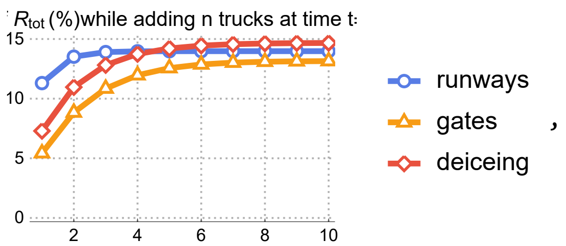
Days	Runways	Gates	Deicing
1	0.960789	0.999868	0.990944
7	0.755784	0.766748	0.746574
30	0.301194	0.0023331	0.0970954

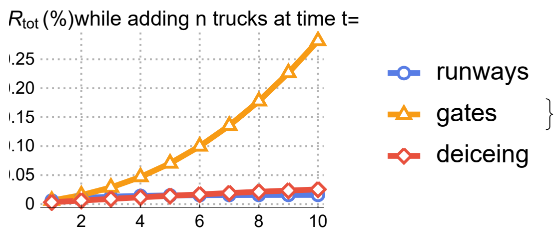
The main cause of failure is dependent on when you examine the system. For T=1, the largest source of failures are the runways as a single runway going down is enough to cause a system failure. For T=7, there’s no main source of failures as all the parts have around the same probability of staying up. For T = 30, gates is the main source of failures as it has a far lower probability of still working compared to runways and deicing stations.

In[162]:=
```
Rtot2[t_] = 100 * (Rkofn[t, 2, n1, λ1] * Rkofn[t, 6, n2, λ2] * Rp[t, n3, λ3]) /. 𝒫λ;
𝒫original = {n1 → 2, n2 → 10, n3 → 2};
𝒫RN3 = {n1 → 3, n2 → 10, n3 → 2};
𝒫GN11 = {n1 → 2, n2 → 11, n3 → 2};
𝒫DN3 = {n1 → 2, n2 → 10, n3 → 3};

R1 = (Rtot2[1] /. 𝒫original);
R7 = (Rtot2[7] /. 𝒫original);
R30 = (Rtot2[30] /. 𝒫original);
Rrunways56[t_] = Table[(Rtot2[t] /. {n1 → 2 + n, n2 → 10, n3 → 2}) - Rt, {n, 1, 10}];
Rgates56[t_] = Table[(Rtot2[t] /. {n1 → 2, n2 → 10 + n, n3 → 2}) - Rt, {n, 1, 10}];
Rdeice56[t_] = Table[(Rtot2[t] /. {n1 → 2, n2 → 10, n3 → 2 + n}) - Rt, {n, 1, 10}];

Table[ListLinePlot[{Rrunways56[t], Rgates56[t], Rdeice56[t]}, PlotRange → All,
  PlotLabel → {"Diff Rtot (%) while adding n trucks at time t=", t },
  AxesLabel → {n, "diff Rtot"}, GridLines → Automatic,
  DataRange → {1, 10}, PlotTheme → "Business",
  PlotLegends → {"runways", "gates", "deiceing"}], {t, {1, 7, 30}}]
Grid[{{"Days", "Rtot (%) 𝒫original", "Diff +1R", "Diff +1G", "Diff +1D" },
  {1, R1, (Rtot2[1] /. 𝒫RN3) - R1, (Rtot2[1] /. 𝒫GN11) - R1, (Rtot2[1] /. 𝒫DN3) - R1},
  {7, R1, (Rtot2[7] /. 𝒫RN3) - R1, (Rtot2[7] /. 𝒫GN11) - R1, (Rtot2[7] /. 𝒫DN3) - R1},
  {30, R30, (Rtot2[30] /. 𝒫RN3) - R30, (Rtot2[30] /. 𝒫GN11) - R30, (Rtot2[30] /. 𝒫DN3) - R30
  }}, Background → {None, {GrayLevel[0.7`], {White}}},
 Dividers → {Black, {2 → Black}}, Frame → True, Spacings → {2, {2, {0.7`}, 2}}]
```

Out[173]=







Out[174]=

Days	Rtot (%) 𝒫original	Diff +1R	Diff +1G	Diff +1D
1	95.1963	3.77003	0.0112117	0.78718
7	95.1963	-40.6286	-46.5049	-44.6399
30	0.00682305	0.00615697	0.00609144	0.00315892

In the plots above we have plotted the difference of R_{tot}(%) in respect to the different times t=1,7,30 days when changing different values of n.

For **t = 1**, the largest improvement comes from adding runways. This is because both deicing stations and gates have forms of redundancy and are therefore unlikely to fail day 1. Do note that runways are modelled as 2 out of n.

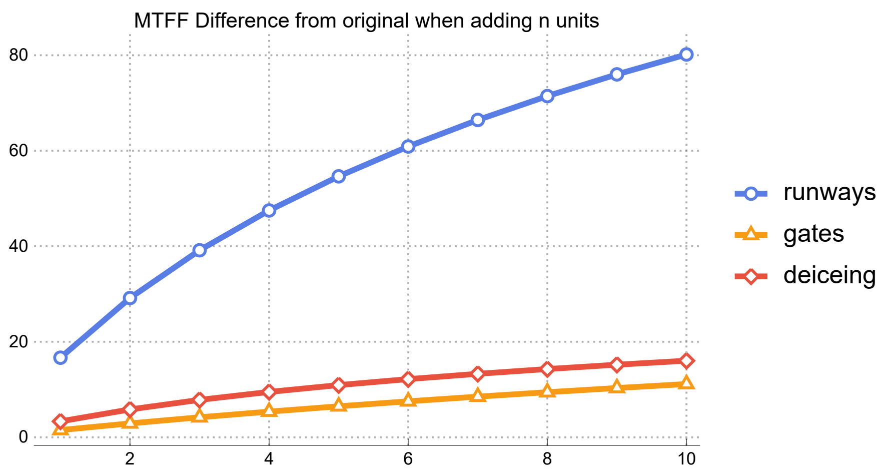
For **t = 7**, runways are still the single strongest increase, but deicing stations are catching up for larger n-values.
For **t = 30**, we can see that the only parameter worth modifying is the number of gates as this yields a relatively large increase for higher values of n. However, while the increase might seem large on the graph, the actual increase is minor at best and resources can most likely be spent better other places than increasing the number of gates.

In[175]:=
```
MTFFR = Integrate[Rkofn[t, 2, 2, λ1] /. 𝒫λ, {t, 0, ∞}];
MTFFG = Integrate[Rkofn[t, 6, 10, λ2] /. 𝒫λ, {t, 0, ∞}];
MTFFD = Integrate[Rkofn[t, 1, 2, λ3] /. 𝒫λ, {t, 0, ∞}];

Grid[{{"diff", "MTFF+1N", "MTFF+2N" , "MTFF+3N" },
  {"Runways",  Integrate[Rkofn[t, 2, 3, λ1] /. 𝒫λ, {t, 0, ∞}] - MTFFR,
   Integrate[Rkofn[t, 2, 4, λ1] /. 𝒫λ, {t, 0, ∞}] - MTFFR,
   Integrate[Rkofn[t, 2, 5, λ1] /. 𝒫λ, {t, 0, ∞}] - MTFFR},
  {"Gates", Integrate[Rkofn[t, 6, 11, λ2] /. 𝒫λ, {t, 0, ∞}] - MTFFG,
   Integrate[Rkofn[t, 6, 12, λ2] /. 𝒫λ, {t, 0, ∞}] - MTFFG,
   Integrate[Rkofn[t, 6, 13, λ2] /. 𝒫λ, {t, 0, ∞}] - MTFFG},
  {"Deice", Integrate[Rkofn[t, 1, 3, λ3] /. 𝒫λ, {t, 0, ∞}] - MTFFD,
   Integrate[Rkofn[t, 1, 4, λ3] /. 𝒫λ, {t, 0, ∞}] - MTFFD,
   Integrate[Rkofn[t, 1, 5, λ3] /. 𝒫λ, {t, 0, ∞}] - MTFFD}
 }, Background → {None, {GrayLevel[0.7`], {White}}}, Dividers → {Black, {2 → Black}},
 Frame → True, Spacings → {2, {2, {0.7`}, 2}}]

MTFFrunways56 =
  Table[Integrate[Rkofn[t, 2, 2 + n, λ1] /. 𝒫λ, {t, 0, ∞}] - MTFFR, {n, 1, 10}];
MTFFgates56 = Table[Integrate[Rkofn[t, 6, 10 + n, λ2] /. 𝒫λ, {t, 0, ∞}] - MTFFG,
   {n, 1, 10}];
MTFFdeice56 = Table[Integrate[Rkofn[t, 1, 2 + n, λ3] /. 𝒫λ, {t, 0, ∞}] - MTFFD,
   {n, 1, 10}];

ListLinePlot[{MTFFrunways56, MTFFgates56, MTFFdeice56}, PlotRange → All,
 PlotLabel → "MTFF Difference from original when adding n units",
 AxesLabel → {n, "diff MTFF from original"}, GridLines → Automatic, DataRange → {1, 10},
 PlotLegends → {"runways", "gates", "deiceing"}, PlotTheme → "Business"]
```

Out[178]=

diff	MTFF+1N	MTFF+2N	MTFF+3N
Runways	16.6667	29.1667	39.1667
Gates	1.51515	2.90404	4.18609
Deice	3.33333	5.83333	7.83333

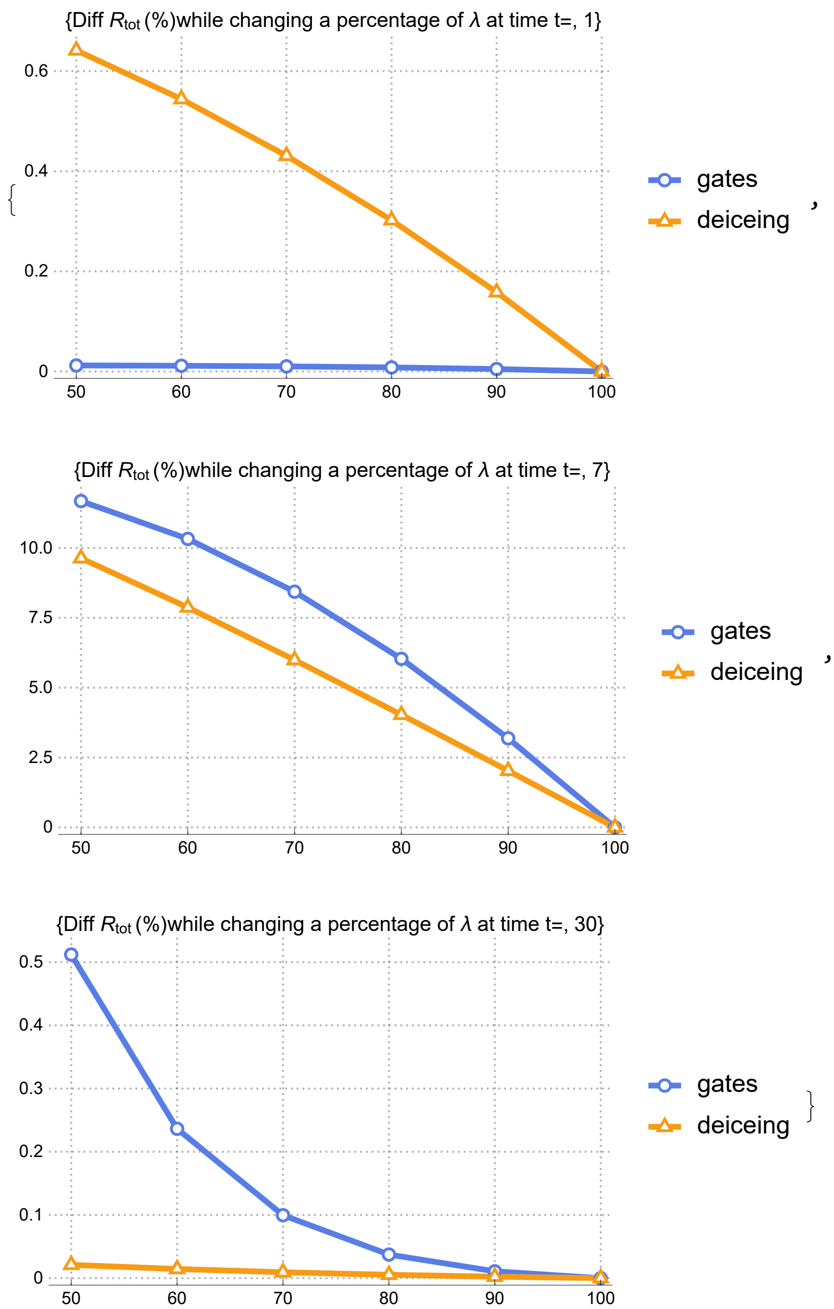
Out[182]=



The reason for the large increases in runway's MTFF is that it is modelled as 2-out-of-n. Adding more runways results in more redundancy and allows the repairmen much more leeway compared to the original 2-out-of-2. It is worth noting that increasing the amount of runways are both expensive and requires a lot of free space and is therefore not as attractive as it can seem on paper.

In[202]:=
```
Rtot3[t_] = 100 * (Rkofn[t, 2, 2, λ1] * Rkofn[t, 6, 10, λ2] * Rp[t, 2, λ3]);

𝒫λ = {λ1 → 0.02, λ2 → 0.06, λ3 → 0.1};
λR1 = (Rtot3[1] /. 𝒫λ);
λR7 = (Rtot3[7] /. 𝒫λ);
λR30 = (Rtot3[30] /. 𝒫λ);

Rgates56λ[t_] = Table[
    (Rtot3[t] /. {λ1 → 0.02, λ2 → (0.06 * i) / 100, λ3 → 0.1}) - λRt, {i, 100, 50, -10}];
Rdeice56λ[t_] = Table[(Rtot3[t] /. {λ1 → 0.02, λ2 → 0.06, λ3 → (0.1 * i) / 100}) - λRt,
    {i, 100, 50, -10}];
Table[ListLinePlot[{Rgates56λ[t], Rdeice56λ[t]}, PlotRange → All,
  PlotLabel → {"Diff Rtot (%) while changing a percentage of λ at time t=", t},
  AxesLabel → {%, "diff Rtot"}, GridLines → Automatic, DataRange → {100, 50},
  PlotTheme → "Business", PlotLegends → {"gates", "deiceing"}], {t, {1, 7, 30}}]
```

Out[209]=



In the plots above we have plotted the difference of R_{tot}(%)(y-value) in respect to the different times t=1,7,30 days when changing the percentage of λ(x-values). λ_{gates}at 100%=0.06 and 0.03 at 50%. λ_{deice}at 100%=0.1 and 0.05 at 50%.

Achieving lower failure rate by improving routines is generally less impactful compared to increasing the number of a given resource. However, this might be a low hanging fruit and can be improved without much investment outside of the time spent optimizing maintenance routines and will still improve airport performance. Runways are omitted from the graphs due to failures being caused by snowfall which is out of the airport's control.