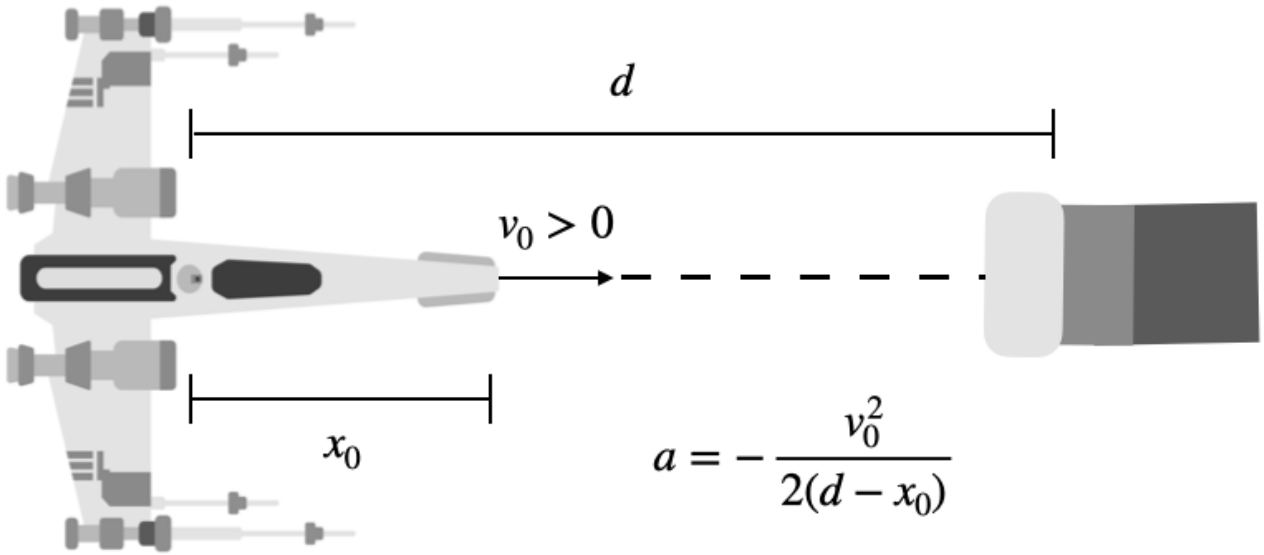
Affine Systems of ODEs in Isabelle/HOL for Hybrid-program Verification

Jonathan Julián Huerta y Munive

University of Sheffield, UK

Verification of Hybrid Systems



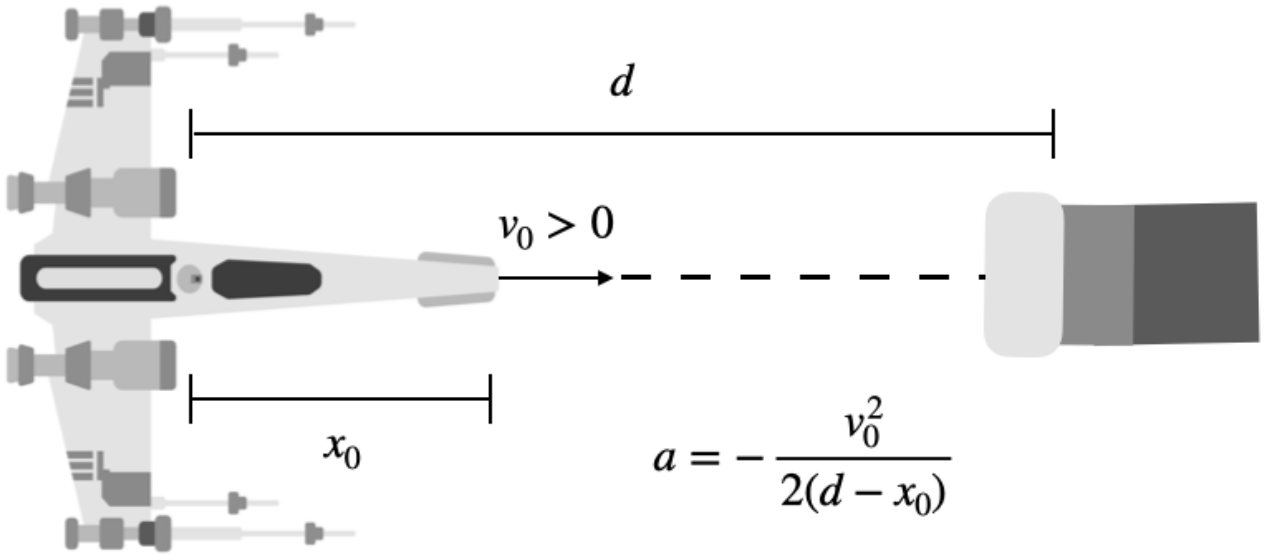
$$\begin{aligned}
\text{dynamics} &= x' = v, v' = a \\
\text{pre} &= x = x_0 \wedge v = v_0 \\
\text{pos} &= v = 0 \leftrightarrow x = d \\
\text{control} &= a := -v_0^2/(2(d - x_0)) \\
\text{dock} &= \text{control}; \text{dynamics} \qquad \text{hybrid program} \\
\{\text{pre}\}\,\text{dock}\,\{\text{pos}\} &\qquad \text{correctness spec}
\end{aligned}$$

Previous Work

- Developed Isabelle/HOL components that
  - benefit from huge libraries of topology, analysis, and ODEs;
  - use algebras for regular programs (modularity);
  - work with partial correctness specifications;
  - are correct by construction.

- They support the following procedure:
  - show that ODEs satisfy Picard-Lindelöf's theorem;
  - supply the solution to the system of ODEs;
  - certify that the solution satisfies the system, and
  - if successful, apply rule for verification of ODEs.

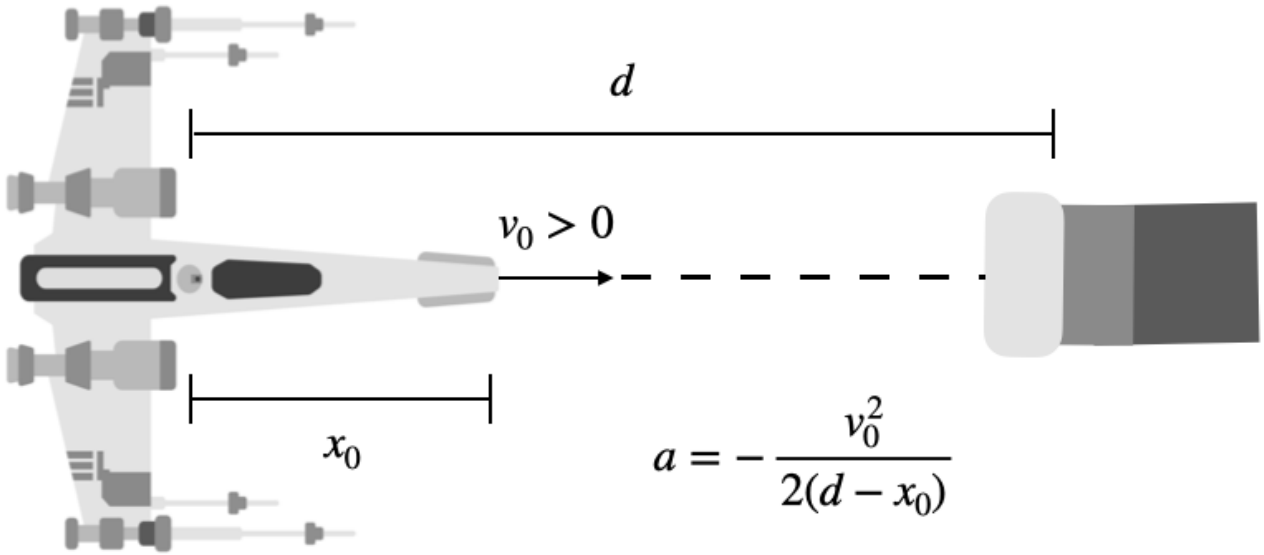
Affine Systems of ODEs



dynamics $\leftrightarrow$ $x' = v, v' = a$

$$\leftrightarrow \begin{pmatrix} x' \\ v' \\ a' \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} x \\ v \\ a \end{pmatrix}.$$

Affine Systems of ODEs



Many models in nature involve affine systems.

$$X' t = A t \cdot X t + B t$$

How does the procedure simplify by formalising affine systems?

Main Contributions

Formalisation of linear and affine systems of ODEs that

1. extends and integrates libraries of linear algebra and ODEs;
2. gives a generic proof that they satisfy Picard-Lindelöf's theorem;
3. provides their general solution in terms of the matrix exponential;
4. includes results for diagonalisation, and
5. simplifies the hybrid program verification procedure.

https://www.isa-afp.org/entries/Matrices_for_ODEs.html

Hybrid Programs

Syntax

$$\alpha, \beta ::= \textbf{skip} \mid \textbf{abort} \mid x := e \mid x' = f \;\&\; G \mid \alpha; \beta \mid \alpha + \beta \mid \alpha^* \mid ?p$$

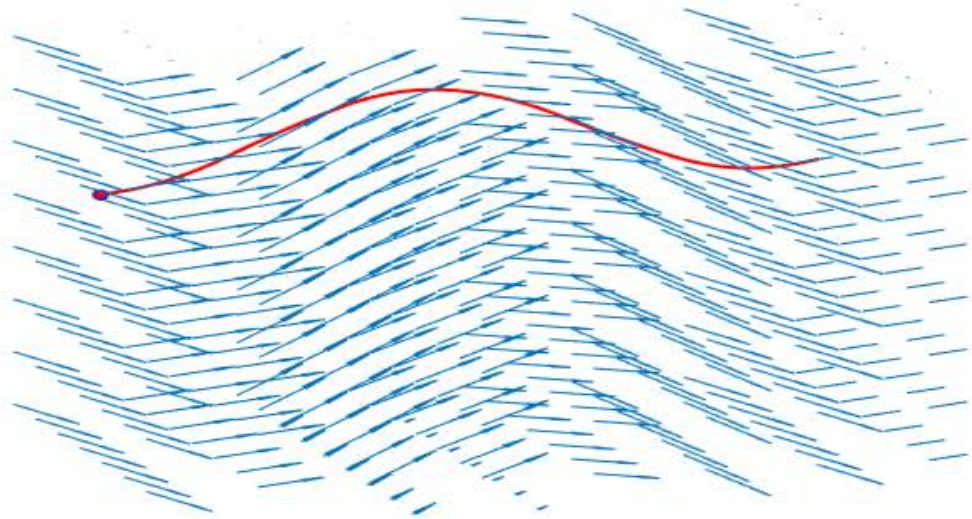
Semantics

- **if** p **then** α **else** $\beta = p\,;\alpha + \neg p\,;\beta$
- $\{p\}\,\alpha\,\{q\} \leftrightarrow p\,;\alpha \leq \alpha\,;q$
- state transformer semantics $S \to \mathcal{P}\,S$
- program states $s \in S$ as functions $s : V \to \mathbb{R}$.
- assignments are $(x := e)\,s = \{s[x \mapsto e\,s]\}$

What about ODEs?

Vector Field

$$X'\ t = f\ t\,(X\ t)$$



where

$$X : T \subseteq \mathbb{R} \to S \qquad f : T \to S \to S \qquad X\,0 = s$$

$$\text{orbit} : s \mapsto \{X\ t \mid t \in T\}$$

Semantics for ODEs

Semantics $S \to \mathcal{P}\, S$

- $\mathsf{Sols}\, f\, T\, s = \{X : T \to S \mid (\forall t \in T.\ X'\, t = f\, t\, (X\, t) \wedge X\, 0 = s\}$
- $\mathsf{orbit}_G^X\, s = \{X\, t \mid t \in T \wedge (\forall \tau \in [0, t].\ G\, (X\, \tau))\}$
- $(x' = f\ \&\ G)\, s = \bigcup \{\mathsf{orbit}_G^X\, s \mid X \in \mathsf{Sols}\, f\, T\, s\}$

easy to obtain if there is a unique solution $X : T \to S$ to the IVPs associated to each s and the vector field f

Verification Rule

$$\frac{\forall s \in S. \exists! X : T \to S.\ X\, 0 = s \wedge (\forall t \in T.\ X'\, t = f\, t\, (X\, t))}{\{\lambda s.\ s \in S \to \forall t \in T.\ (\forall \tau \in [0, t].\ G\, (X\, t)) \to q\, (X\, t)\}\, x' = f\ \&\ G\, \{q\}}$$

Semantics for ODEs

Semantics $S \to \mathcal{P}\, S$

- $\mathsf{Sols}\, f\, T\, s = \{X : T \to S \mid (\forall t \in T.\ X'\, t = f\, t\, (X\, t) \wedge X\, 0 = s\}$
- $\mathsf{orbit}_G^X\, s = \{X\, t \mid t \in T \wedge (\forall \tau \in [0, t].\ G\, (X\, \tau))\}$
- $(x' = f\ \&\ G)\, s = \bigcup \{\mathsf{orbit}_G^X\, s \mid X \in \mathsf{Sols}\, f\, T\, s\}$

easy to obtain if there is a unique solution $X : T \to S$ to the IVPs associated to each s and the vector field f

Verification Rule

$$\frac{\text{True for } X'\, t = A\, t \cdot (X\, t) + B\, t}{\{\lambda s.\ s \in S \to \forall t \in T.\ (\forall \tau \in [0, t].\ G\, (X\, t)) \to q\, (X\, t)\}\, x' = f\ \&\ G\, \{q\}}$$

Verification of Affine Systems

1. Procedure for affine system $X' t = A t \cdot X t + B t$ where $A t$ is a square matrix and $B t$ is a vector.

   1.1 show that ODEs satisfy Picard-Lindelöf's theorem;

   1.2 supply the solution to the system of ODE;

   1.3 certify that the solution satisfies the system;

   1.4 if successful, apply rule for verification of ODEs.

Verification of Affine Systems

1. Procedure for affine system $X' t = A t \cdot X t + B t$ where $A t$ is a square matrix and $B t$ is a vector.

   1.1 ~~show that ODEs satisfy Picard-Lindelöf's theorem;~~
   - affine systems satisfy existence and uniqueness

   1.2 supply the solution to the system of ODE;

   1.3 certify that the solution satisfies the system;

   1.4 if successful, apply rule for verification of ODEs.

Verification of Affine Systems

1. Procedure for affine system $X' t = A t \cdot X t + B t$ where $A t$ is a square matrix and $B t$ is a vector.

   1.1 ~~show that ODEs satisfy Picard-Lindelöf's theorem;~~
   - affine systems satisfy existence and uniqueness

   1.2 ~~supply the solution to the system of ODE;~~
   - if $A t = A$ and $B t = B$, then

$$X t = \exp(tA) \cdot s + \exp(tA) \cdot \int_0^t (\exp(-\tau A) \cdot B)$$

   1.3 certify that the solution satisfies the system;

   1.4 if successful, apply rule for verification of ODEs.

Verification of Affine Systems

1. Procedure for affine system $X' t = A t \cdot X t + B t$ where $A t$ is a square matrix and $B t$ is a vector.

   1.1 ~~show that ODEs satisfy Picard-Lindelöf's theorem;~~
   - affine systems satisfy existence and uniqueness

   1.2 ~~supply the solution to the system of ODE;~~
   - if $A t = A$ and $B t = B$, then

$$X t = \exp(tA) \cdot s + \exp(tA) \cdot \int_0^t (\exp(-\tau A) \cdot B)$$

   1.3 ~~certify that the solution satisfies the system~~;
   - we can show this in Isabelle/HOL

   1.4 if successful, apply rule for verification of ODEs.

Verification of Affine Systems

1. Procedure for affine system $X' t = A t \cdot X t + B t$ where $A t$ is a square matrix and $B t$ is a vector.

   1.1 ~~show that ODEs satisfy Picard-Lindelöf's theorem;~~
   - affine systems satisfy existence and uniqueness

   1.2 ~~supply the solution to the system of ODE;~~
   - if $A t = A$ and $B t = B$, then

$$X t = \exp(tA) \cdot s + \exp(tA) \cdot \int_0^t (\exp(-\tau A) \cdot B)$$

   1.3 ~~certify that the solution satisfies the system~~;
   - we can show this in Isabelle/HOL

   1.4 if successful, apply rule for verification of ODEs.
   - we need to handle matrix exponentials

Formalising Existence and Uniqueness

lemma *picard-lindeloef-affine*:
fixes $A :: real \Rightarrow real\hat{}'n\hat{}'n$
assumes *Ahyp*: *matrix-continuous-on* T A
and $\bigwedge \tau\ \varepsilon.\ \tau \in T \Longrightarrow \varepsilon > 0 \Longrightarrow$ *bdd-above* $\{\|A\ t\|_{op} \mid t.\ dist\ \tau\ t \leq \varepsilon\}$
and *Bhyp*: *continuous-on* T B **and** *open* S
and $t_0 \in T$ **and** *Thyp*: *open* T **and** *is-interval* T
shows *picard-lindeloef* $(\lambda t\ s.\ A\ t *v s + B\ t)\ T\ S\ t_0$
⟨*proof*⟩

Issues

- operator norm for matrices $\|A\|_{op} = Sup\ \{\|A *v x\| \mid x.\ \|x\| = 1\}$
- continuity in the space of matrices

Notation

- type of vectors $s :: real\hat{}'n$ and matrices $A :: real\hat{}'n\hat{}'n$
- matrix times vector is $A *v s$

Formalising Solutions of Affine Systems

lemma *autonomous-affine-sol-is-exp-plus-int*:
assumes $D\ X = (\lambda t.\ A *_V X\ t + B)\ on\ \{t_0 -- t\}$ **and** $X\ t_0 = s$
shows $X\ t = exp\ ((t - t_0) *_R A) *_V s +$
$exp\ ((t - t_0) *_R A) *_V (\int_{t0}^{t} (exp\ (-(\tau - t_0) *_R A) *_V B) \partial\tau)$
$\langle proof \rangle$

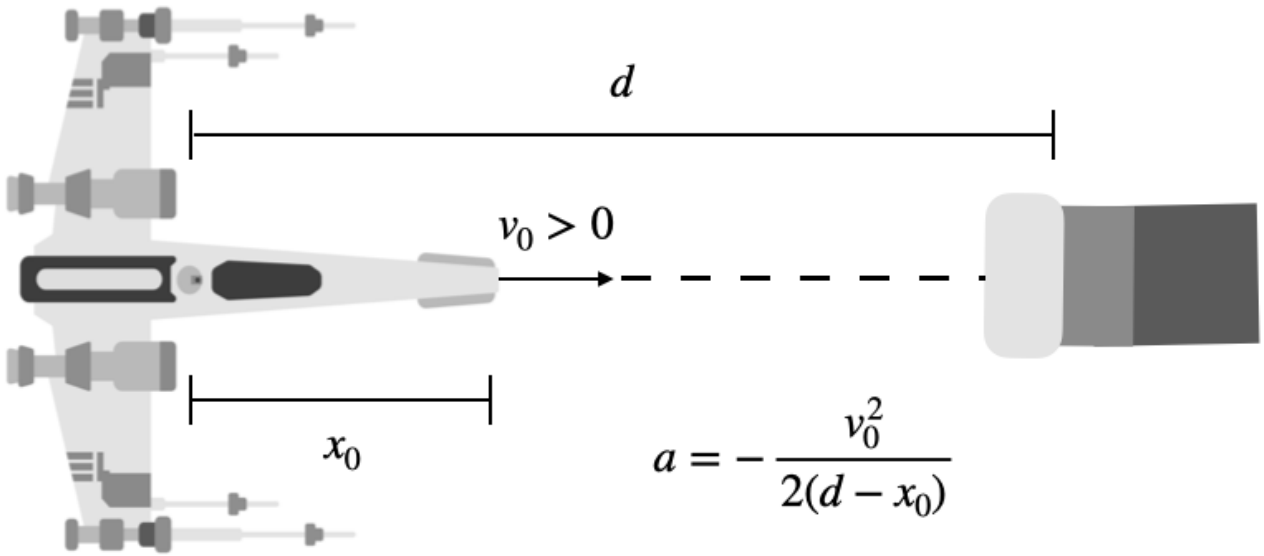
Issues

- exp defined for *real-normed-algebra-1* with 1 such that $\|1\| = 1$.
- Define **typedef** *'m sq-mtx* = *UNIV*::(*real^'m^'m*) *set*
- Prove **instance** *sq-mtx* :: (*finite*) *real-normed-algebra-1* $\langle proof \rangle$
- Prove **instance** *sq-mtx* :: (*finite*) *banach* $\langle proof \rangle$
- Redefine and reprove using $\|A\|_{op}$

Notation

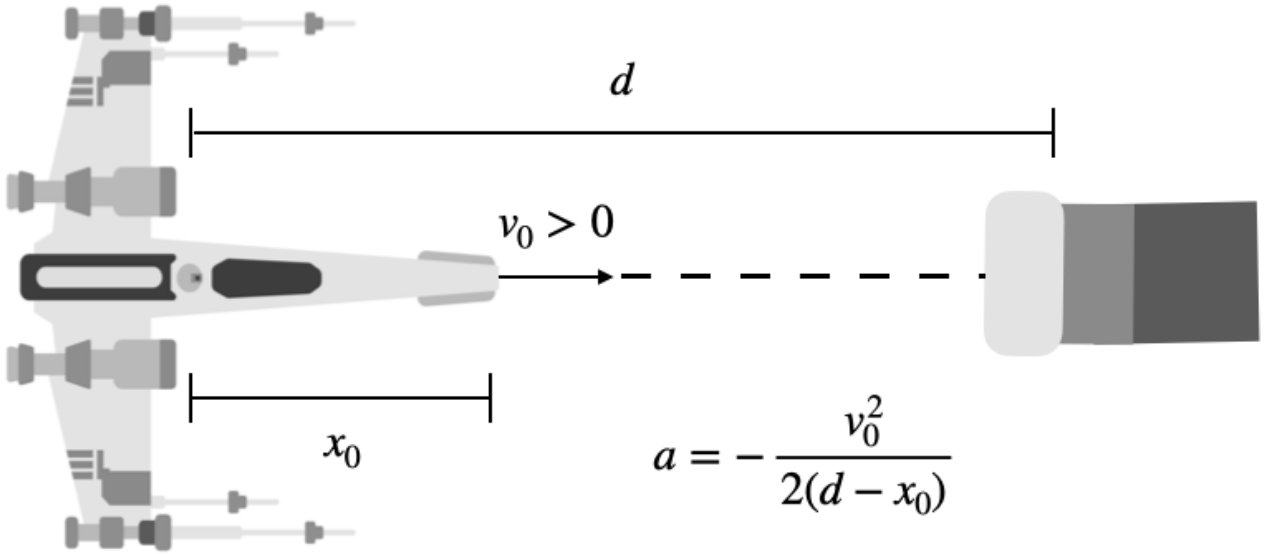
- scaling of vector is $a *_R s$ and ith coordinate of s is $s\ \$\ i$
- derivatives $D\ X = (\lambda t.\ f\ t\ (X\ t))\ on\ T$

Verification Example



abbreviation *mtx-cnst-acc* :: *3 sq-mtx* (*K*)
where *K* ≡ *mtx* (
[*0*,*1*,*0*] #
[*0*,*0*,*1*] #
[*0*,*0*,*0*] # [])

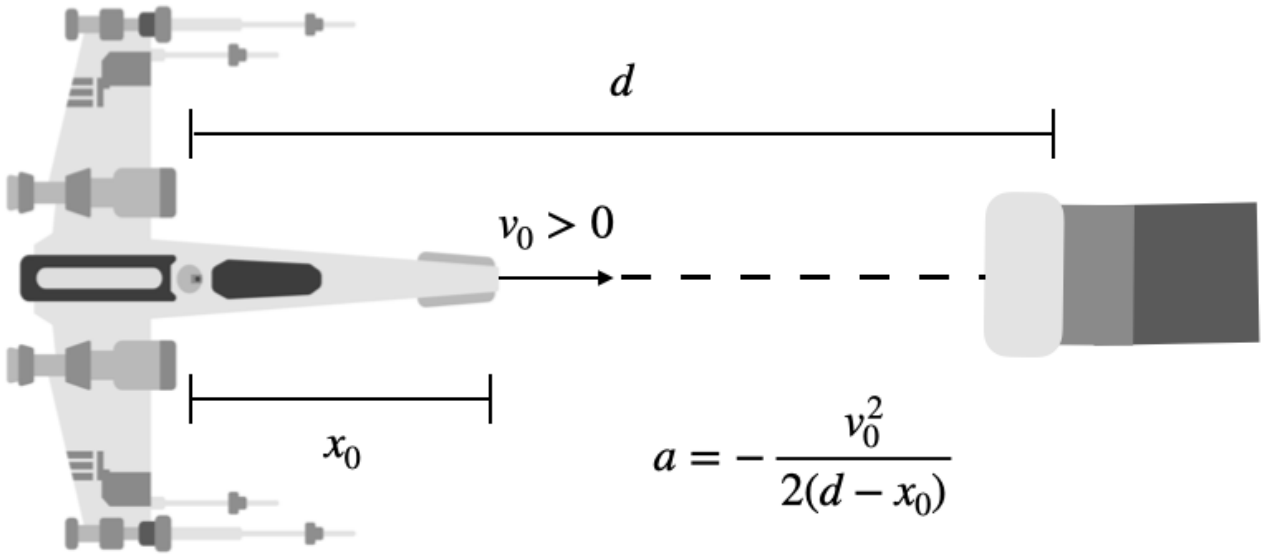
Verification Example



lemma *exp-mtx-cnst-acc*: $exp\ (t *_R K) = ((t *_R K)^2 /_R 2) + (t *_R K) + 1$
unfolding *exp-def* **apply**(*subst suminf-eq-sum*[*of 2*])
using *powN-scaleR-mtx-cnst-acc* **by** (*simp-all add*: *numeral-2-eq-2*)

lemma *exp-mtx-cnst-acc-vec-mult-eq*: $exp\ (t *_R K) *_V s =$
$vector\ [s\$3 * t\hat{}2/2 + s\$2 * t + s\$1, s\$3 * t + s\$2, s\$3]$
apply(*subst exp-mtx-cnst-acc*, *subst pow2-scaleR-mtx-cnst-acc*)
apply(*simp add*: *sq-vec-mult-eq vector-def*)
unfolding *UNIV-3* **by** (*simp add*: *fun-eq-iff*)

Verification Example



lemma *docking-station*:

assumes $d > x_0$ **and** $v_0 > 0$

shows *PRE* $(\lambda s.\ s\$1 = x_0 \wedge s\$2 = v_0)$

HP $((3 ::= (\lambda s.\ -(v_0\ \hat{}\ 2/(2*(d-x_0)))));\ x' = (*v)\ K\ \&\ G)$

POST $(\lambda s.\ s\$2 = 0 \longleftrightarrow s\$1 = d)$

apply(*clarsimp simp*: *le-fun-def local-flow.fbox-g-ode*[*OF local-flow-sq-mtx-linear*[*of K*]])

unfolding *exp-mtx-cnst-acc-vec-mult-eq* **using** *assms* **by** (*simp add*: *docking-station-arith*)

Handling Exponentials

General solution involving $\exp(tA)$ is easier for diagonalisable matrices

- A is diagonalisable if it is similar to a diagonal matrix

$$\begin{pmatrix} \iota_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \iota_n \end{pmatrix}$$

- A is similar to B if there is P such that $A = P^{-1}BP$
- If A is diagonalisable, then

$$\exp(tA) = P^{-1} \begin{pmatrix} \exp(t\iota_1) & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \exp(t\iota_n) \end{pmatrix} P$$

Handling Exponentials

lemma *mtx-invertible-def*:
mtx-invertible A $\longleftrightarrow$ $(\exists\, A'.\ A' * A = 1 \wedge A * A' = 1)$
⟨*proof*⟩

definition *similar-sq-mtx* :: $('n::finite)$ *sq-mtx* $\Rightarrow$ $'n$ *sq-mtx* $\Rightarrow$ *bool*
where $(A \sim B) \longleftrightarrow (\exists\ P.$ *mtx-invertible P* $\wedge\ A = P^{-1} * B * P)$

definition *diag-mat f* $= (\chi\ i\ j.$ *if i* $= j$ *then f i else 0*$)$

lemma *exp-scaleR-diagonal1*:
assumes *mtx-invertible P* **and** $A = P^{-1} * (\mathrm{diag}\ i.\ f\ i) * P$
shows *exp* $(t *_R A) = P^{-1} * (\mathrm{diag}\ i.$ *exp* $(t * f\ i)) * P$
⟨*proof*⟩

Example: Certifying a Diagonalisation

- Problem: diagonalise

```
abbreviation mtx-hOsc :: real ⇒ real ⇒ 2 sq-mtx (A)
  where A a b ≡ mtx
   ([0, 1] #
    [a, b] # [])
```

- CAS supplies change of basis P and eigenvalues ι_1 and ι_2

```
abbreviation mtx-chB-hOsc :: real ⇒ real ⇒ 2 sq-mtx (P)
  where P a b ≡ mtx
   ([a, b] #
    [1, 1] # [])
```

Example: Certifying a Diagonalisation

- certify diagonlisation

lemma *mtx-hOsc-diagonalisable*:
defines $\iota_1 \equiv (b - sqrt\ (b\hat{}2+4*a))/2$ **and** $\iota_2 \equiv (b + sqrt\ (b\hat{}2+4*a))/2$
assumes $b^2 + a * 4 > 0$ **and** $a \neq 0$
shows $A\ a\ b =$
$P\ (-\iota_2/a)\ (-\iota_1/a) * (\mathrm{diag}\ i.\ if\ i = 1\ then\ \iota_1\ else\ \iota_2) * (P\ (-\iota_2/a)\ (-\iota_1/a))^{-1}$
⟨*proof*⟩

- simplify diagonalisation of exponential

lemma *mtx-hOsc-solution-eq*:
defines $\iota_1 \equiv (b - sqrt\ (b^2+4*a))/2$ **and** $\iota_2 \equiv (b + sqrt\ (b^2+4*a))/2$
defines $\Phi\ t \equiv mtx\ ($
$[\iota_2*exp(t*\iota_1) - \iota_1*exp(t*\iota_2),\quad exp(t*\iota_2)-exp(t*\iota_1)]\#$
$[a*exp(t*\iota_2) - a*exp(t*\iota_1),\ \iota_2*exp(t*\iota_2)-\iota_1*exp(t*\iota_1)]\#[])$
assumes $b^2 + a * 4 > 0$ **and** $a \neq 0$
shows $(1/sqrt\ (b^2 + a*4)) *_R (\Phi\ t) =$
$P\ (-\iota_2/a)\ (-\iota_1/a)*(\mathrm{diag}\ i.\ exp\ (t*(if\ i{=}1\ then\ \iota_1\ else\ \iota_2)))*(P\ (-\iota_2/a)\ (-\iota_1/a))^{-1}$
⟨*proof*⟩

Conclusions

- New mathematical components for affine and linear systems
  - that extend libraries of linear algebra and ODEs,
  - remove certifications of framework for verification of hybrid systesms,
  - extend automatic tactics of the framework.
- Future work:
  - Connect to libraries of Jordan normal form with code generation.
  - Do a larger case study with these improvements.
  - Formalise general solution for non-autonomous case with resolvent matrix.
  - Integrate with a CAS that supplies solutions and diagonalisations, leaving the certification work to Isabelle.

https://www.isa-afp.org/entries/Matrices_for_ODEs.html