Two Theorems on Hermitian Matrices

Sage Binder and Zilin Jiang

November 25, 2024

Abstract

We formalize two results on Hermitian matrices. First, Sylvester's criterion: a hermitian matrix is positive definite if and only if all its leading principal submatrices have positive determinant. Second, Cauchy's eigenvalue interlacing theorem: given a principal submatrix B of a hermitian matrix A, the eigenvalues of B interlace those of A.

Our approach to Sylvester's criterion is fairly standard, and required us to formalize Schur's block matrix determinant formula, which gives a formula for the determinant of a block matrix (A, B, C, D) when A is invertible.

Our approach to Cauchy's eigenvalue interlacing theorem follows a proof given in a set of lecture notes by Dr. David Bindel [1]. This approach involved formalizing the Courant-Fischer minimax theorem (a theorem about the Rayleigh quotient, which we define in this entry). In our statement of the Courant-Fischer minimax theorem, we refer to the infimum and supremum instead of the minimum and maximum, as this simplifies the proof and is sufficient to prove Cauchy's eigenvalue interlacing theorem.

Contents

theory *Misc-Matrix-Results*
imports *Commuting-Hermitian.Commuting-Hermitian*
BenOr-Kozen-Reif.More-Matrix
Jordan-Normal-Form.Spectral-Radius
Jordan-Normal-Form.DL-Rank-Submatrix
Jordan-Normal-Form.Jordan-Normal-Form-Uniqueness
Jordan-Normal-Form.VS-Connect
QHLProver.Complex-Matrix
Fishers-Inequality.Matrix-Vector-Extras
Complex-Bounded-Operators.Extra-Jordan-Normal-Form
Hermite-Lindemann.Misc-HLW
begin

hide-type (**open**) *Matrix-Legacy.mat*
hide-const (**open**) *Matrix-Legacy.mat*
hide-fact (**open**) *Finite-Cartesian-Product.mat-def*
hide-const (**open**) *Finite-Cartesian-Product.mat*
hide-fact (**open**) *Matrix-Legacy.mat-def*
hide-const (**open**) *Finite-Cartesian-Product.row*
hide-fact (**open**) *Finite-Cartesian-Product.row-def*
hide-const (**open**) *Matrix-Legacy.row*
hide-fact (**open**) *Matrix-Legacy.row-def*

hide-const (**open**) *Matrix-Legacy.col*
hide-fact (**open**) *Matrix-Legacy.col-def*
hide-const (**open**) *Determinants.det*
hide-fact (**open**) *Determinants.det-def*
hide-type (**open**) *Finite-Cartesian-Product.vec*
hide-const (**open**) *Finite-Cartesian-Product.vec*
hide-fact (**open**) *Finite-Cartesian-Product.vec-def*
hide-type (**open**) *Matrix-Legacy.vec*
hide-const (**open**) *Matrix-Legacy.vec*
hide-fact (**open**) *Matrix-Legacy.vec-def*
hide-const (**open**) *Coset.order*
hide-fact (**open**) *Coset.order-def*
hide-const (**open**) *Linear-Algebra.adjoint*
hide-fact (**open**) *Linear-Algebra.adjoint-def*
hide-const (**open**) *Finite-Cartesian-Product.transpose*
hide-fact (**open**) *Finite-Cartesian-Product.transpose-def*
unbundle *no-inner-syntax*
unbundle *no-vec-syntax*
hide-const (**open**) *Missing-List.span*
hide-const (**open**)
  dependent
  independent
  real-vector.representation
  real-vector.subspace
  span
  real-vector.extend-basis
  real-vector.dim
hide-const (**open**) *orthogonal*
no-notation *fps-nth* (**infixl** $ *75*)

1 Determinant, Invertible, and Eigenvalue Lemmas

definition *eigvals-of* [*simp*]:
  eigvals-of M es ⟷ *char-poly M* = (∏ *a*←*es*. [:− *a*, *1*:]) ∧ *length es* = *dim-row M*

lemma *det-is-prod-of-eigenvalues*:
  fixes *A* :: *complex mat*
  assumes *square-mat A*
  shows *det A* = (∏ *e* ← (*eigvals A*). *e*)
⟨*proof*⟩

lemma *eigvals-of-spectrum*:
  (*A*::(*complex mat*)) ∈ *carrier-mat n n* ⟹ *eigvals-of A* α ⟹ *spectrum A* = *set* α
  ⟨*proof*⟩

lemma *trivial-kernel-imp-nonzero-eigenvalues*:
fixes M :: $'a$::{*idom*,*ring-1-no-zero-divisors*} *mat*
assumes *square-mat* M
assumes *mat-kernel* $M \subseteq \{0_v\ (\textit{dim-row}\ M)\}$
assumes *eigenvalue* M e
shows $e \neq 0$
⟨*proof*⟩

lemma *trivial-kernel-imp-invertible*:
fixes M :: *complex mat*
assumes *square-mat* M
assumes *mat-kernel* $M \subseteq \{0_v\ (\textit{dim-row}\ M)\}$
shows *invertible-mat* M
⟨*proof*⟩

lemma *trivial-kernel-imp-det-nz*:
fixes M :: *complex mat*
assumes *square-mat* M
assumes *mat-kernel* $M \subseteq \{0_v\ (\textit{dim-row}\ M)\}$
shows *det* $M \neq 0$
⟨*proof*⟩

lemma *similar-mats-eigvals*:
assumes $A \in$ *carrier-mat* n n
assumes $B \in$ *carrier-mat* n n
assumes *similar-mat* A B
assumes *eigvals-of* A *es*
shows *eigvals-of* B *es*
⟨*proof*⟩

lemma *scale-eigvals*:
fixes A :: *complex mat*
assumes $A \in$ *carrier-mat* n n
assumes $B = c \cdot_m A$
assumes *eigvals-of* A *es*
shows *eigvals-of* B (*map* ($\lambda x.\ c * x$) *es*)
⟨*proof*⟩

lemma *neg-mat-eigvals*:
fixes A :: *complex mat*
assumes $A \in$ *carrier-mat* n n
assumes *eigvals-of* A *es*
shows *eigvals-of* $(-A)$ (*rev* (*map* ($\lambda x.\ -x$) *es*))
⟨*proof*⟩

2 Quadratic Form

definition *quadratic-form* :: $'a$ *mat* $\Rightarrow$ $'a$ *vec* $\Rightarrow$ $'a$::{*conjugatable-ring*} **where**
quadratic-form M $x \equiv$ *inner-prod* x $(M *_v x)$

abbreviation *QF* ≡ *quadratic-form*

lemma *hermitian-quadratic-form-real*:
fixes *A* :: *complex mat*
fixes *v* :: *complex vec*
assumes *A* ∈ *carrier-mat n n*
assumes *v* ∈ *carrier-vec n*
assumes *hermitian A*
shows *QF A v* ∈ *Reals*
⟨*proof*⟩

declare
quadratic-form-def[*simp*]

3 Leading Principal Submatrix Lemmas

definition *leading-principal-submatrix* :: *'a mat* ⇒ *nat* ⇒ *'a mat* **where**
[*simp*]: *leading-principal-submatrix A k* = *submatrix A* {..<*k*} {..<*k*}

abbreviation *lps* ≡ *leading-principal-submatrix*

lemma *leading-principal-submatrix-carrier*:
$m \geq n \Longrightarrow A \in$ *carrier-mat m m* $\Longrightarrow$ *lps A n* ∈ *carrier-mat n n*
⟨*proof*⟩

lemma *pick-n*:
assumes $i \leq n$
shows *pick* {..*n*} *i* = *i*
⟨*proof*⟩

lemma *pick-n-le*:
assumes $i < n$
shows *pick* {..<*n*} *i* = *i*
⟨*proof*⟩

lemma *leading-principal-submatrix-index*:
assumes *A* ∈ *carrier-mat n n*
assumes $k \leq n$
assumes $i < k$
assumes $j < k$
shows (*lps A k*)$$(*i*,*j*) = *A*$$(*i*,*j*)
⟨*proof*⟩

lemma *nested-leading-principle-submatrices*:
assumes *A* ∈ *carrier-mat n n*
assumes $k_1 \leq k_2$
assumes $k_2 \leq n$
shows *lps A* k_1 = *lps* (*lps A* k_2) k_1 (**is** *?lhs* = *?rhs*)

⟨*proof*⟩

4 Submatrix Lemmas

lemma *submatrix-as-matrix-prod*:
  fixes A :: *complex mat*
  assumes $A \in$ *carrier-mat* n n
  assumes $I \subseteq \{..<n\}$
  assumes $I \neq \{\}$
  defines $m \equiv$ *card* I
  defines $B \equiv$ *submatrix* A I I
  defines *u-cols-inds* $\equiv$ *map* (*pick* I) [*0*..<m]
  defines *u-cols* $\equiv$ *map* ((!) (*unit-vecs* n)) *u-cols-inds*
  defines (*Inm*::*complex mat*) $\equiv$ *mat-of-cols* n *u-cols*
  defines (*Inm*′::*complex mat*) $\equiv$ Inm^H
  shows $B = Inm' * A * Inm$
    $Inm' * Inm = 1_m\ m$
    Inm $\in$ *carrier-mat* n m
    inj-on $((*_v)$ *Inm*) (*carrier-vec* m)
⟨*proof*⟩

lemma *submatrix-as-matrix-prod-obt*:
  fixes A :: *complex mat*
  assumes $A \in$ *carrier-mat* n n
  assumes $I \subseteq \{..<n\}$
  assumes $I \neq \{\}$
  defines $m \equiv$ *card* I
  defines $B \equiv$ *submatrix* A I I
  obtains *Inm* **where** $B = Inm^H * A * Inm$
    $Inm^H * Inm = 1_m\ m$
    Inm $\in$ *carrier-mat* n m
    inj-on $((*_v)$ *Inm*) (*carrier-vec* m)
  ⟨*proof*⟩

5 Hermitian and Conjugate Lemmas

lemma *hermitian-is-square*: *hermitian* $A \Longrightarrow$ *square-mat* A
  ⟨*proof*⟩

lemma *hermitian-eigenvalues-real*:
  assumes (A::(*complex mat*)) $\in$ *carrier-mat* n n
  assumes *hermitian* A
  assumes *eigenvalue* A e
  shows $e \in$ *Reals*
  ⟨*proof*⟩

lemma *hermitian-spectrum-real*:
  (A::(*complex mat*)) $\in$ *carrier-mat* n n $\Longrightarrow$ *hermitian* A $\Longrightarrow$ *spectrum* $A \subseteq$ *Reals*

⟨*proof*⟩

lemma *leading-principal-submatrix-hermitian*:
assumes $A \in$ *carrier-mat n n*
assumes *hermitian A*
assumes $k \leq n$
shows *hermitian* (*lps A k*) (**is** *hermitian ?A′*)
⟨*proof*⟩

lemma *conjugate-mat-dist*:
fixes *A B* :: *′a::conjugatable-ring mat*
assumes $A \in$ *carrier-mat m n*
assumes $B \in$ *carrier-mat n p*
shows (*conjugate A*) $*$ (*conjugate B*) = *conjugate* ($A * B$)
⟨*proof*⟩

lemma *conjugate-mat-inv*:
fixes *A* :: *′a::{conjugatable-ring,semiring-1} mat*
assumes $A \in$ *carrier-mat n n*
assumes $A' \in$ *carrier-mat n n*
assumes *inverts-mat A A′*
shows *inverts-mat* (*conjugate A*) (*conjugate A′*)
⟨*proof*⟩

lemma *hermitian-mat-inv*:
assumes $A \in$ *carrier-mat n n*
assumes $A' \in$ *carrier-mat n n*
assumes *hermitian A*
assumes *inverts-mat A A′*
shows *hermitian A′*
⟨*proof*⟩

lemma *hermitian-ij-ji*:
hermitian A
⟷ *square-mat A* ∧ (∀ *i j*. *i* < *dim-row A* ∧ *j* < *dim-row A* ⟶ *A*$$(*i*,*j*) = *conjugate* (*A*$$(*j*,*i*)))
⟨*proof*⟩

lemma *negative-hermitian*:
assumes $A \in$ *carrier-mat n n*
assumes *hermitian A*
shows *hermitian* ($-A$)
⟨*proof*⟩

lemma *principal-submatrix-hermitian*:
assumes $A \in$ *carrier-mat n n*
assumes *hermitian A*
assumes $I \subseteq \{..<n\}$
shows *hermitian* (*submatrix A I I*) (**is** *hermitian ?B*)

⟨*proof*⟩

lemma *conjugate-dist-mult-mat*:
  fixes A :: $'a$::*conjugatable-ring mat*
  assumes $A \in$ *carrier-mat m n* $B \in$ *carrier-mat n p*
  shows *conjugate* $(A * B) =$ *conjugate* $A *$ *conjugate* B
    (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *conjugate-dist-add-mat*:
  fixes A :: $'a$::*conjugatable-ring mat*
  assumes $A \in$ *carrier-mat m n* $B \in$ *carrier-mat m n*
  shows *conjugate* $(A + B) =$ *conjugate* $A +$ *conjugate* B
    (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *mat-row-conj*:
  assumes $A \in$ *carrier-mat m n*
  assumes $i < m$
  shows *conjugate* (*row* A i) = *row* (*conjugate* A) i
  ⟨*proof*⟩

lemma *conj-mat-vec-mult*:
  fixes A :: $'a$::{*conjugate*,*conjugatable-ring*} *mat*
  fixes v :: $'a$ *vec*
  assumes $A \in$ *carrier-mat n n*
  assumes $v \in$ *carrier-vec n*
  shows *conjugate* $(A *_v v) =$ (*conjugate* A) $*_v$ (*conjugate* v)
    (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *hermitian-row-col*:
  assumes $A \in$ *carrier-mat n n*
  assumes *hermitian* A
  assumes $i < n$
  shows *row* A i = *conjugate* (*col* A i)
  ⟨*proof*⟩

lemma *hermitian-real-diag-decomp-eigvals*:
  fixes A :: *complex mat*
  assumes $A \in$ *carrier-mat n n*
  assumes *hermitian* A
  assumes *eigvals-of* A *es*
  obtains B U **where**
    real-diag-decomp A B U
    diag-mat B = *es*
    set es $\subseteq$ *Reals*
    $B \in$ *carrier-mat n n*
    $U \in$ *carrier-mat n n*

⟨*proof*⟩

lemma *conjugate-vec-first*:
  assumes $v \in carrier\text{-}vec\ n$
  assumes $i \leq n$
  shows $conjugate\ (vec\text{-}first\ v\ i) = vec\text{-}first\ (conjugate\ v)\ i$
  ⟨*proof*⟩

lemma *conjugate-vec-last*: $i \leq dim\text{-}vec\ v \Longrightarrow conjugate\ (vec\text{-}last\ v\ i) = vec\text{-}last\ (conjugate\ v)\ i$
  ⟨*proof*⟩

lemma *adjoint-is-conjugate-transpose*: $A^H = adjoint\ A$
  ⟨*proof*⟩

lemma *cscalar-prod-symm-conj*:
  $dim\text{-}vec\ (x::('a::\{comm\text{-}semiring\text{-}0,conjugatable\text{-}ring\}\ vec)) = dim\text{-}vec\ (y::'a\ vec)$
    $\Longrightarrow x \bullet c\ y = conjugate\ (y \bullet c\ x)$
  ⟨*proof*⟩

6 Block Matrix Lemmas

lemma *block-mat-vec-mult*:
  fixes x
  assumes $A \in carrier\text{-}mat\ nr1\ nc1$
  assumes $B \in carrier\text{-}mat\ nr1\ nc2$
  assumes $C \in carrier\text{-}mat\ nr2\ nc1$
  assumes $D \in carrier\text{-}mat\ nr2\ nc2$
  assumes $M = four\text{-}block\text{-}mat\ A\ B\ C\ D$
  assumes $x \in carrier\text{-}vec\ (nc1 + nc2)$
  defines $x_1 \equiv vec\text{-}first\ x\ nc1$
  defines $x_2 \equiv vec\text{-}last\ x\ nc2$
  shows $M *_v x = (A *_v x_1 + B *_v x_2)\ @_v\ (C *_v x_1 + D *_v x_2)$
  ⟨*proof*⟩

lemma *mat-vec-prod-leading-principal-submatrix*:
  fixes $A :: ('a :: comm\text{-}ring)\ mat$
  assumes $A \in carrier\text{-}mat\ (Suc\ n)\ (Suc\ n)$
  assumes $x \in carrier\text{-}vec\ (Suc\ n)$
  defines $A_n \equiv lps\ A\ n$
  defines $v_n \equiv vec\text{-}first\ (col\ A\ n)\ n$
  defines $w_n \equiv vec\text{-}first\ (row\ A\ n)\ n$
  defines $a \equiv A\ \$\$\ (n, n)$
  defines $x_n \equiv vec\text{-}first\ x\ n$
  defines $b \equiv x\$n$
  shows $A *_v x = (A_n *_v x_n + b \cdot_v v_n)\ @_v\ (vec\ 1\ (\lambda i.\ (w_n \cdot x_n) + a * b))$ (**is** $?lhs = ?rhs$)
⟨*proof*⟩

lemma *vec-first-index*: $n \leq$ *dim-vec* $v \Longrightarrow i < n \Longrightarrow v\$i =$ (*vec-first* v n)$\$i$
⟨*proof*⟩

lemma *vec-last-index*:
$n \leq$ *dim-vec* $v \Longrightarrow i \in \{$*dim-vec* $v - m..<m\} \Longrightarrow v\$i =$ (*vec-last* v m)$\$(i -$ (*dim-vec* $v - m$))
⟨*proof*⟩

lemma *inner-prod-append*:
assumes $x \in$ *carrier-vec* (*dim-vec* (u $@_v$ v))
shows $x \cdot c$ (u $@_v$ v) = (*vec-first* x (*dim-vec* u)) $\cdot c$ u + (*vec-last* x (*dim-vec* v)) $\cdot c$ v
(u $@_v$ v) $\cdot c$ $x = u \cdot c$ (*vec-first* x (*dim-vec* u)) + $v \cdot c$ (*vec-last* x (*dim-vec* v))
⟨*proof*⟩

6.1 Schur's Formula

proposition *schur-formula*:
fixes M :: *'a::field mat*
assumes $(A,B,C,D) =$ *split-block* M r c
assumes $r <$ *dim-row* M
assumes $c <$ *dim-col* M
assumes *square-mat* M
assumes *square-mat* A
assumes *inverts-mat* A' A
assumes A'-*dim*: $A' \in$ *carrier-mat* r r
shows *det* M = *det* A $*$ *det* $(D - C * A' * B)$
⟨*proof*⟩

7 Positive Definite Lemmas

definition *positive-definite* **where**
positive-definite $M \longleftrightarrow$ *hermitian* M
$\wedge$ ($\forall x \in$ *carrier-vec* (*dim-col* M). $x \neq 0_v$ (*dim-col* M) $\longrightarrow$ *QF* M $x > 0$)

lemma *leading-principal-submatrix-positive-definite*:
fixes A :: *'a::{conjugatable-field,ord} mat*
assumes $A \in$ *carrier-mat* n n
assumes *positive-definite* A
assumes $k \leq n$
shows *positive-definite* (*lps* A k)
⟨*proof*⟩

lemma *positive-definite-invertible*:
fixes M :: *complex mat*
assumes *positive-definite* M
shows *invertible-mat* M
⟨*proof*⟩

lemma *positive-definite-det-nz*:
  fixes *A* :: *complex mat*
  assumes *positive-definite A*
  shows *det A* ≠ *0*
  ⟨*proof*⟩

end
theory *Sylvester-Criterion*
  imports *Misc-Matrix-Results*

begin

8 Sylvester's Criterion Setup

definition *sylvester-criterion* :: (*'a*::{*comm-ring-1*,*ord*}) *mat* ⇒ *bool* **where**
  sylvester-criterion A ⟷ (∀ *k* ∈ {*0*..*dim-row A*}. *Determinant.det* (*lps A k*) > *0*)

lemma *leading-principle-submatrix-sylvester*:
  assumes *A* ∈ *carrier-mat n n*
  assumes *m* ≤ *n*
  assumes *sylvester-criterion A*
  shows *sylvester-criterion* (*lps A m*)
  ⟨*proof*⟩

lemma *sylvester-criterion-positive-det*:
  assumes *A* ∈ *carrier-mat n n*
  assumes *sylvester-criterion A*
  shows *det A* > *0*
⟨*proof*⟩

9 Sylvester's Criterion

9.1 Forward Implication

lemma *sylvester-criterion-forward*:
  fixes *A* :: *complex mat*
  assumes *A* ∈ *carrier-mat n n*
  assumes *x* ∈ *carrier-vec n*
  assumes *hermitian A*
  assumes *sylvester-criterion A*
  assumes $x \neq 0_v$ *n*
  shows *Re* (*QF A x*) > *0*
  ⟨*proof*⟩

9.2 Reverse Implication

lemma *prod-list-gz*:
  fixes *l* :: *real list*

assumes $\forall x \in set\ l.\ x > 0$
shows *prod-list* $l > 0$
⟨*proof*⟩

lemma *sylvester-criterion-reverse*:
fixes A :: *complex mat*
assumes $A \in$ *carrier-mat* $n\ n$
assumes *hermitian* A
assumes *positive-definite* A
shows *sylvester-criterion* A
⟨*proof*⟩

9.3 Theorem Statement

theorem *sylvester-criterion*:
fixes A :: *complex mat*
assumes $A \in$ *carrier-mat* $n\ n$
assumes *hermitian* A
shows *sylvester-criterion* $A \longleftrightarrow$ *positive-definite* A
⟨*proof*⟩

end
theory *Cauchy-Eigenvalue-Interlacing*
imports *Misc-Matrix-Results*

begin

10 Rayleigh Quotient Lemmas

definition *rayleigh-quotient-complex* (ϱ_c) **where**
$\varrho_c\ M\ x = (QF\ M\ x)\ /\ (x \cdot c\ x)$

definition *rayleigh-quotient* (ϱ) **where**
$\varrho\ M\ x = Re\ (\varrho_c\ M\ x)$

declare
rayleigh-quotient-complex-def[*simp*]
rayleigh-quotient-def[*simp*]

lemma *rayleigh-quotient-negative*: $A \in$ *carrier-mat* $n\ n \Longrightarrow x \in$ *carrier-vec* $n \Longrightarrow$
$\varrho\ A\ x = -\ \varrho\ (-\ A)\ x$
⟨*proof*⟩

lemma *rayleigh-quotient-complex-scale*:
fixes k :: *real*
assumes $A \in$ *carrier-mat* $n\ n$
assumes $v \in$ *carrier-vec* n
assumes $k \neq 0$
shows $\varrho_c\ A\ v = \varrho_c\ A\ (k \cdot_v\ v)$

⟨*proof*⟩

lemma *rayleigh-quotient-scale*:
fixes k :: *real*
assumes $A \in$ *carrier-mat* n n
assumes $v \in$ *carrier-vec* n
assumes $k \neq 0$
shows $\varrho\ A\ v = \varrho\ A\ (k \cdot_v v)$
⟨*proof*⟩

lemma *hermitian-rayleigh-quotient-real*:
fixes A :: *complex mat*
assumes $A \in$ *carrier-mat* n n
assumes $v \in$ *carrier-vec* n
assumes *hermitian* A
assumes $v \neq 0_v\ n$
shows $\varrho_c\ A\ v \in$ *Reals*
⟨*proof*⟩

11 Vector Summation Lemmas

lemma *complex-vec-norm-sum*:
fixes x :: *complex vec*
assumes $x \in$ *carrier-vec* n
shows *vec-norm* x = *csqrt* $((\sum i \in \{..<n\}.\ (cmod\ (x\$i))\hat{\ }2))$
⟨*proof*⟩

lemma *inner-prod-vec-sum*:
assumes $v \in$ *carrier-vec* n
assumes $w \in$ *carrier-vec* n
assumes $B \subseteq$ *carrier-vec* n
assumes *finite* B
assumes v = *finsum-vec* $TYPE('a::conjugatable\text{-}ring)\ n\ (\lambda b.\ cs\ b \cdot_v b)\ B$
shows *inner-prod* $w\ v = (\sum b \in B.\ cs\ b * inner\text{-}prod\ w\ b)$
⟨*proof*⟩

lemma *sprod-vec-sum*:
assumes $v \in$ *carrier-vec* n
assumes $w \in$ *carrier-vec* n
assumes $B \subseteq$ *carrier-vec* n
assumes *finite* B
assumes v = *finsum-vec* $TYPE('a::\{comm\text{-}ring\})\ n\ (\lambda b.\ cs\ b \cdot_v b)\ B$
shows $w \cdot v = (\sum b \in B.\ cs\ b * (w \cdot b))$
⟨*proof*⟩

lemma *mat-vec-mult-sum*:
assumes $v \in$ *carrier-vec* n
assumes $A \in$ *carrier-mat* n n
assumes $B \subseteq$ *carrier-vec* n

assumes *finite B*
assumes $v = \textit{finsum-vec}\ \textit{TYPE}('a{::}\textit{comm-ring})\ n\ (\lambda b.\ cs\ b \cdot_v b)\ B$
shows $A *_v v = \textit{finsum-vec}\ \textit{TYPE}('a{::}\textit{comm-ring})\ n\ (\lambda b.\ cs\ b \cdot_v (A *_v b))\ B$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

12 Module Span Lemmas

context *module*
begin

lemma *mk-coeffs-of-list*:
assumes $\alpha \in (\textit{set}\ A \to \textit{carrier}\ R)$
shows $\exists\, c \in \{0..<\textit{length}\ A\} \to \textit{carrier}\ R.\ \forall\, v \in \textit{set}\ A.\ \textit{mk-coeff}\ A\ c\ v = \alpha\ v$
⟨*proof*⟩

lemma *span-list-span*:
assumes $\textit{set}\ A \subseteq \textit{carrier}\ M$
shows $\textit{span-list}\ A = \textit{span}\ (\textit{set}\ A)$
⟨*proof*⟩

end

13 Module Homomorphism Linear Combination and Span Lemmas

context *mod-hom*
begin

lemma *lincomb-list-distrib*:
assumes $\textit{set}\ S \subseteq \textit{carrier}\ M$
assumes $\alpha \in \{..<\textit{length}\ S\} \to \textit{carrier}\ R$
shows $f\ (M.\textit{lincomb-list}\ \alpha\ S) = N.\textit{lincomb-list}\ \alpha\ (\textit{map}\ f\ S)$
⟨*proof*⟩

lemma *lincomb-distrib*:
assumes *inj-on f S*
assumes $S \subseteq \textit{carrier}\ M$
assumes $\alpha \in S \to \textit{carrier}\ R$
assumes $\forall\, v \in S.\ \alpha\ v = \beta\ (f\ v)$
assumes *finite S*
shows $f\ (M.\textit{lincomb}\ \alpha\ S) = N.\textit{lincomb}\ \beta\ (f`S)$
⟨*proof*⟩

lemma *lincomb-distrib-obtain*:
assumes *inj-on f S*
assumes $S \subseteq \textit{carrier}\ M$
assumes $\alpha \in S \to \textit{carrier}\ R$

assumes $\forall\, v \in S.\ \alpha\ v = \beta\ (f\ v)$
assumes *finite S*
obtains β **where** $(\forall\, v \in S.\ \alpha\ v = \beta\ (f\ v)) \wedge f\ (M.lincomb\ \alpha\ S) = N.lincomb\ \beta$ $(f`S)$
⟨*proof*⟩

lemma *image-span-list*:
assumes *set vs* $\subseteq$ *carrier M*
shows *f*‘(*M.span-list vs*) = *N.span-list* (*map f vs*) (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *image-span*:
assumes *finite vs*
assumes *vs* $\subseteq$ *carrier M*
shows *f*‘(*M.span vs*) = *N.span* (*f*‘*vs*)
⟨*proof*⟩

end

14 Linear Map Lemmas

lemma (**in** *linear-map*) *inj-image-lin-indpt*:
assumes *inj-on T* (*carrier V*)
assumes *S* $\subseteq$ *carrier V*
assumes *V.module.lin-indpt S*
assumes *finite S*
shows *W.module.lin-indpt* (*T*‘*S*)
⟨*proof*⟩

lemma *linear-map-mat*:
assumes *A* $\in$ *carrier-mat n m*
shows *linear-map class-ring* (*module-vec TYPE*('*a*::{*field*,*ring-1*}) *m*) (*module-vec TYPE*('*a*) *n*) (($*_v$) *A*)
(**is** *linear-map ?K ?V ?W ?T*)
⟨*proof*⟩

15 Courant-Fischer Minimax Theorem

We follow the proof given in this set of lecture notes by Dr. David Bindel: https://www.cs.cornell.edu/courses/cs6210/2019fa/lec/2019-11-04.pdf.

definition *sup-defined* :: '*a*::*preorder set* ⇒ *bool* **where**
sup-defined S ⟷ *S* ≠ {} ∧ *bdd-above S*

definition *inf-defined* :: '*a*::*preorder set* ⇒ *bool* **where**
inf-defined S ⟷ *S* ≠ {} ∧ *bdd-below S*

locale *hermitian-mat* = *complex-vec-space n* **for** *n* +
fixes *A* :: *complex mat*

assumes *dim-is*: $A \in$ *carrier-mat n n*
assumes *is-herm*: *hermitian A*
begin

definition *dimensional* :: *complex vec set* $\Rightarrow$ *nat* $\Rightarrow$ *bool* **where**
dimensional $\mathcal{V}$ *k* $\longleftrightarrow$ ($\exists$ *vs*. $\mathcal{V}$ = *span vs* $\wedge$ *card vs* = *k* $\wedge$ *vs* $\subseteq$ *carrier-vec n* $\wedge$ *lin-indpt vs*)

lemma *dimensional-n*: *dimensional* $\mathcal{V}$ *k* $\Longrightarrow$ $\mathcal{V} \subseteq$ *carrier-vec n*
⟨*proof*⟩

lemma *dimensional-n-vec*: $\bigwedge$*v*. *v* $\in \mathcal{V}$ $\Longrightarrow$ *dimensional* $\mathcal{V}$ *k* $\Longrightarrow$ *v* $\in$ *carrier-vec n*
⟨*proof*⟩

Note here that we refer to the Inf and Sup rather than the Min and Max.

definition *rayleigh-min*:
rayleigh-min $\mathcal{V}$ = *Inf* {ϱ *A v* | *v*. *v* $\neq$ 0_v *n* $\wedge$ *v* $\in \mathcal{V}$ $\wedge$ *vec-norm v* = *1*}

definition *rayleigh-max*:
rayleigh-max $\mathcal{V}$ = *Sup* {ϱ *A v* | *v*. *v* $\neq$ 0_v *n* $\wedge$ *v* $\in \mathcal{V}$ $\wedge$ *vec-norm v* = *1*}

definition *maximin* :: *nat* $\Rightarrow$ *real* **where**
maximin k = *Sup* {*rayleigh-min* $\mathcal{V}$ | $\mathcal{V}$. *dimensional* $\mathcal{V}$ *k*}

definition *minimax* :: *nat* $\Rightarrow$ *real* **where**
minimax k = *Inf* {*rayleigh-max* $\mathcal{V}$ | $\mathcal{V}$. *dimensional* $\mathcal{V}$ (*n* − *k* + *1*)}

definition *maximin-defined* **where**
maximin-defined k $\longleftrightarrow$ *sup-defined* {*rayleigh-min* $\mathcal{V}$ | $\mathcal{V}$. *dimensional* $\mathcal{V}$ *k*}

definition *minimax-defined* **where**
minimax-defined k $\longleftrightarrow$ *inf-defined* {*rayleigh-max* $\mathcal{V}$ | $\mathcal{V}$. *dimensional* $\mathcal{V}$ (*n* − *k* + *1*)}

end

locale *courant-fischer* = *hermitian-mat n* **for** *n* +
fixes Λ *U* :: *complex mat*
fixes *es* :: *complex list*
assumes *eigvals*: *eigvals-of A es*
assumes *eigvals-sorted*: *sorted-wrt* ($\geq$) *es*
assumes *A-decomp*: *real-diag-decomp A* Λ *U*
$\wedge$ *diag-mat* Λ = *es*
$\wedge$ *set es* $\subseteq$ *Reals*
$\wedge$ *U* $\in$ *carrier-mat n n*
$\wedge$ $\Lambda \in$ *carrier-mat n n*
begin

sublocale *conjugatable-vec-space TYPE*(*complex*) *n* ⟨*proof*⟩

lemma *dim*: *local.dim* = *n*
⟨*proof*⟩

lemma *fin-dim*: *fin-dim* ⟨*proof*⟩

lemma *gr-n-lin-dpt*:
assumes $B \subseteq$ *carrier-vec n*
assumes *card* $B >$ *local.dim*
shows *lin-dep B*
⟨*proof*⟩

lemma *rayleigh-kx*:
assumes $v \in$ *carrier-vec n*
assumes $k \neq 0$
assumes $v \neq 0_v\ n$
shows $\varrho\ A\ (k \cdot_v v) = \varrho\ A\ v$
⟨*proof*⟩

lemma *unit-vec-rayleigh-formula*:
assumes *unit-v*: *vec-norm* $v = 1$
assumes *v-dim*: $v \in$ *carrier-vec n*
shows $\varrho\ A\ v = (\sum j \in \{..<n\}.\ es!j * (cmod\ ((U^H *_v v)\$j))\hat{}2)$
⟨*proof*⟩

lemma *rayleigh-bdd-below'*:
assumes $k \leq n$
shows $\exists m. \forall v \in$ *carrier-vec n*. $v \neq 0_v\ n \longrightarrow \varrho\ A\ v \geq m$
⟨*proof*⟩

lemma *rayleigh-bdd-below*:
assumes *dimensional* $\mathcal{V}$ *k*
assumes $k \leq n$
shows $\exists m. \forall v \in \mathcal{V}.\ v \neq 0_v\ n \longrightarrow \varrho\ A\ v \geq m$
⟨*proof*⟩

lemma *rayleigh-min-exists*:
assumes *dimensional* $\mathcal{V}$ *k*
assumes $k \leq n$
shows $\forall x \in \{\varrho\ A\ v \mid v.\ v \neq 0_v\ n \wedge v \in \mathcal{V} \wedge$ *vec-norm* $v = 1\}.$ *rayleigh-min* $\mathcal{V}$
$\leq x$
⟨*proof*⟩

lemma *courant-fischer-unit-rayleigh-helper2*:
assumes *dimensional* $\mathcal{V}$ $(k + 1)$
shows $\exists v.$ *vec-norm* $v = 1 \wedge v \in \mathcal{V} \wedge v \neq 0_v\ n \wedge \varrho\ A\ v \leq es!k$
⟨*proof*⟩

lemma *courant-fischer-unit-rayleigh-helper3*:

assumes $n > 0$
assumes $k < n$
assumes *eigvals-of A es*
defines *es-R* $\equiv$ *map Re es*
shows $\exists \mathcal{V}$. *dimensional* $\mathcal{V}$ $(k + 1) \wedge (\forall v.\ v \neq 0_v\ n \wedge v \in \mathcal{V} \wedge$ *vec-norm* $v = 1$
$\longrightarrow$ *es-R* ! $k \leq \varrho\ A\ v)$
⟨*proof*⟩

theorem *courant-fischer-maximin*:
assumes $n > 0$
assumes $k < n$
shows *es*!k = *maximin* $(k + 1)$
maximin-defined $(k + 1)$
⟨*proof*⟩

end

lemma *courant-fischer-maximin*:
fixes A :: *complex mat*
assumes $n > 0$
assumes $k < n$
assumes $A \in$ *carrier-mat n n*
assumes *hermitian A*
assumes *eigvals-of A es*
assumes *sorted-wrt* $(\geq)$ *es*
shows *es*!k = *hermitian-mat.maximin n A* $(k + 1)$ *hermitian-mat.maximin-defined*
n A $(k + 1)$
⟨*proof*⟩

lemma *maximin-minimax*:
fixes A :: *complex mat*
assumes $A \in$ *carrier-mat n n*
assumes *hermitian A*
assumes $k < n$
shows *hermitian-mat.maximin n* $(-A)$ $(n - k)$ = $-$ *hermitian-mat.minimax n*
A $(k + 1)$
hermitian-mat.maximin-defined n $(-A)$ $(n - k) \Longrightarrow$ *hermitian-mat.minimax-defined*
n A $(k + 1)$
⟨*proof*⟩

lemma *courant-fischer-minimax*:
fixes A :: *complex mat*
assumes $n > 0$
assumes $k < n$
assumes $A \in$ *carrier-mat n n*
assumes *hermitian A*
assumes *eigvals-of A es*
assumes *sorted-wrt* $(\geq)$ *es*
shows *es*!k = *hermitian-mat.minimax n A* $(k + 1)$

hermitian-mat.minimax-defined n A (k + 1)
⟨*proof*⟩

15.1 Theorem Statement

theorem *courant-fischer*:
fixes *A* :: *complex mat*
assumes $n > 0$
assumes $k < n$
assumes $A \in$ *carrier-mat n n*
assumes *hermitian A*
assumes *eigvals-of A es*
assumes *sorted-wrt* (≥) *es*
shows *es*!*k* = *hermitian-mat.minimax n A (k + 1)*
es!*k* = *hermitian-mat.maximin n A (k + 1)*
hermitian-mat.minimax-defined n A (k + 1)
hermitian-mat.maximin-defined n A (k + 1)
⟨*proof*⟩

16 Cauchy Eigenvalue Interlacing Theorem

We follow the proof given in this set of lecture notes by Dr. David Bindel: https://www.cs.cornell.edu/courses/cs6210/2019fa/lec/2019-11-04.pdf

16.1 Theorem Statement and Proof

theorem *cauchy-eigval-interlacing*:
fixes *A W* :: *complex mat*
assumes $n > 0$
assumes $j < n$
assumes $m \leq n$
assumes $m > 0$
assumes $j < m$

assumes $A \in$ *carrier-mat n n*
assumes *hermitian A*
assumes *eigvals-of A* α
assumes *sorted-wrt* (≥) α

assumes $W \in$ *carrier-mat n m*
assumes $W^H * W = 1_m\ m$
assumes *inj-on* $(\lambda v.\ W *_v v)$ (*carrier-vec m*)

defines $B \equiv W^H * A * W$
assumes *eigvals-of B* β
assumes *sorted-wrt* (≥) β
shows $\alpha!(n-m+j) \leq \beta!j$ $\beta!j \leq \alpha!j$
⟨*proof*⟩

corollary *cauchy-eigval-interlacing-alt*:
  fixes A W :: *complex mat*
  assumes $n > 0$
  assumes $j < n$
  assumes $m \leq n$
  assumes $m > 0$
  assumes $j < m$

  assumes $A \in$ *carrier-mat* n n
  assumes *hermitian* A
  assumes *eigvals-of* A α
  assumes *sorted-wrt* $(\geq)$ α

  assumes $W \in$ *carrier-mat* n m
  assumes $W^H * W = 1_m$ m
  assumes *inj-on* $(\lambda v.\ W *_v v)$ (*carrier-vec* m)

  defines $B \equiv W^H * A * W$
  assumes *eigvals-of* B β
  assumes *sorted-wrt* $(\geq)$ β

  shows $\beta!j \in \{\alpha!(n-m+j)..\alpha!j\}$
  ⟨*proof*⟩

16.2 Principal Submatrix Corollaries

corollary *ps-eigval-interlacing*:
  fixes A :: *complex mat*
  fixes k
  assumes $n > 0$
  assumes $A \in$ *carrier-mat* n n
  assumes *hermitian* A
  assumes *eigvals-of* A α
  assumes *sorted-wrt* $(\geq)$ α

  assumes $I \subseteq \{..<n\}$
  assumes $I \neq \{\}$
  defines $B \equiv$ *submatrix* A I I
  defines $m \equiv$ *card* I
  assumes *eigvals-of* B β
  assumes *sorted-wrt* $(\geq)$ β

  assumes $j < m$
  shows $\alpha!(n-m+j) \leq \beta!j$ $\beta!j \leq \alpha!j$
⟨*proof*⟩

corollary *ps-eigval-interlacing-alt*:
  fixes A :: *complex mat*

fixes *k*
assumes $n > 0$
assumes $A \in$ *carrier-mat n n*
assumes *hermitian A*
assumes *eigvals-of A* α
assumes *sorted-wrt* $(\geq)$ α

assumes $I \subseteq \{..<n\}$
assumes $I \neq \{\}$
defines $B \equiv$ *submatrix A I I*
defines $m \equiv$ *card I*
assumes *eigvals-of B* β
assumes *sorted-wrt* $(\geq)$ β

assumes $j < m$
shows $\beta!j \in \{\alpha!(n-m+j)..\alpha!j\}$
⟨*proof*⟩

16.3 Leading Principal Submatrix Corollaries

corollary *lps-eigval-interlacing*:
fixes *A* :: *complex mat*
fixes *k*
assumes $n > 0$
assumes $A \in$ *carrier-mat n n*
assumes *hermitian A*
assumes *eigvals-of A* α
assumes *sorted-wrt* $(\geq)$ α

assumes $0 < m$
assumes $m \leq n$
defines $B \equiv$ *lps A m*
assumes *eigvals-of B* β
assumes *sorted-wrt* $(\geq)$ β

assumes $j < m$
shows $\alpha!(n-m+j) \leq \beta!j$ $\beta!j \leq \alpha!j$
⟨*proof*⟩

corollary *lps-eigval-interlacing-alt*:
fixes *A* :: *complex mat*
fixes *k*
assumes $n > 0$
assumes $A \in$ *carrier-mat n n*
assumes *hermitian A*
assumes *eigvals-of A* α
assumes *sorted-wrt* $(\geq)$ α

assumes $0 < m$

assumes $m \leq n$
defines $B \equiv lps\ A\ m$
assumes *eigvals-of* $B\ \beta$
assumes *sorted-wrt* $(\geq)\ \beta$

assumes $j < m$
shows $\beta!j \in \{\alpha!(n-m+j)..\alpha!j\}$
⟨*proof*⟩

end

References

[1] D. Bindel. Lecture notes. https://www.cs.cornell.edu/courses/cs6210/2019fa/lec/2019-11-04.pdf, 2019. CS6210 at Cornell University.