One-sided compact representability gives GSS rigidity

`MathlibAnnex.Analysis.CStarAlgebra.nonempty_starAlgEquiv_iff_nonempty_geometricStructure_homeomorph_of_exists_isCompactlyRepresented`

theorem

Obtains algebraic rigidity when either side admits a faithful compact action.

Statement

Let A, B be complex C*-algebras. For a complex C*-algebra D, write $S(D)$ for its geometric structure space. Its points are the unions of kernels of norm-one bounded complex-linear functionals supporting a maximal proper face of the closed unit ball of D; faces use real convexity. Write c_D for its hull–kernel closure:

$$c_D(E) = \left\{ I \in S(D) : \bigcap_{J \in E} J \subseteq I \right\}, \qquad E \subseteq S(D).$$

These are the geometric structure definition and its closure, with the norm-one description justified by the norm-one support characterization. Let $\mathcal{K}(H)$ denote the algebra of compact operators on a complex Hilbert space H.

Assume that at least one of A, B admits a faithful $*$-representation π on a complex Hilbert space H whose image is contained in $\mathcal{K}(H)$. Then A and B are $*$-isomorphic if and only if $S(A)$ and $S(B)$ are homeomorphic as closure spaces. Here a closure-space homeomorphism is a bijection $e : S(A) \to S(B)$ satisfying

$$e(c_A(E)) = c_B(e(E)), \qquad E \subseteq S(A).$$

The equality is required for every subset, not just finite subsets.

Assumptions

A faithful compact-valued representation is required on at least one side. All Hilbert spaces are complex and complete. The algebras and representation spaces may be zero. Representations need not be unital; no separability or countability assumption is imposed.

Conclusion

The equivalence concerns existence of a $*$-algebra isomorphism and existence of a closure-space homeomorphism. It does not require the isomorphism to induce a prescribed homeomorphism, or assert uniqueness.

Main citations

The either-side compact rigidity theorem is the stated equivalence. Rigidity for a given compact action allows an independent Hilbert universe and uses the canonical compact c0-decomposition. The existential representation comparison links the existential condition to the same reduced atomic action.

Lean source signature (exact)

Exact source and declaration. Only the literal signature is displayed here; the linked source retains the proof.

```
theorem
nonempty_starAlgEquiv_iff_nonempty_geometricStructure_homeomorph_of_exists_isCompactlyRepresent
ed
    (hCompact :
      (∃ (K : Type u) (normed : NormedAddCommGroup K),
        letI : NormedAddCommGroup K := normed
        ∃ inner : InnerProductSpace ℂ K,
          letI : InnerProductSpace ℂ K := inner
          ∃ complete : CompleteSpace K,
            letI : CompleteSpace K := complete
            ∃ pi : NonUnitalCStarRepresentation A K, pi.IsCompactlyRepresented) ∨
      (∃ (K : Type v) (normed : NormedAddCommGroup K),
        letI : NormedAddCommGroup K := normed
        ∃ inner : InnerProductSpace ℂ K,
          letI : InnerProductSpace ℂ K := inner
          ∃ complete : CompleteSpace K,
            letI : CompleteSpace K := complete
            ∃ pi : NonUnitalCStarRepresentation B K, pi.IsCompactlyRepresented)) :
    Nonempty (A ≃⋆ₐ[ℂ] B) ↔
      Nonempty (ClosureSpace.Homeomorph (NormedSpace.geometricStructureClosure ℂ A)
        (NormedSpace.geometricStructureClosure ℂ B))
```

In the source	Mathematical meaning

Two C*-algebras
`variable {A : Type u} {B : Type v}`
`variable [NonUnitalCStarAlgebra A]`
`[NonUnitalCStarAlgebra B]`
	The inputs A, B are arbitrary complex C*-algebras, including zero. Their size levels are independent. The nonunital representation type imposes no requirement that either algebra lack a unit.

Representation space on the A side
`(∃ (K : Type u) (normed : NormedAddCommGroup K),`
`letI : NormedAddCommGroup K := normed`
`∃ inner : InnerProductSpace ℂ K,`
`letI : InnerProductSpace ℂ K := inner`
	The left alternative requires a space with a norm and compatible complex inner product. The source calls it `K`; it is the space H of the Statement. Each `letI` makes the chosen structure on that same space available in the next type.

Complete faithful compact action on A
`∃ complete : CompleteSpace K,`
`letI : CompleteSpace K := complete`
`∃ pi : NonUnitalCStarRepresentation A K,`
`pi.IsCompactlyRepresented) ∨`
	The next existential requires completeness of that space. Its `letI` makes the Hilbert-space representation type available. There must then exist a faithful $\pi : A \to \mathcal{B}(H)$ with $\pi(A) \subseteq \mathcal{K}(H)$. The final disjunction permits the alternative on the B side.

In the source	Mathematical meaning

Representation space on the B side
`(∃ (K : Type v) (normed : NormedAddCommGroup K),`
`  letI : NormedAddCommGroup K := normed`
`  ∃ inner : InnerProductSpace ℂ K,`
`  letI : InnerProductSpace ℂ K := inner`
	The right alternative has its own existentially quantified normed space and compatible complex inner product, again installed by `letI`. It is not required to be the representation space of the A alternative.

Complete faithful compact action on B
`∃ complete : CompleteSpace K,`
`  letI : CompleteSpace K := complete`
`  ∃ pi : NonUnitalCStarRepresentation B K,`
`pi.IsCompactlyRepresented)) :`
	This space must be complete, and must admit a faithful $\pi : B \to \mathcal{B}(H)$ with $\pi(B) \subseteq \mathcal{K}(H)$. The two complete representation conditions are alternatives, not simultaneous requirements.

Meaning of compact representability
`def`
`NonUnitalCStarRepresentation.IsCompactlyRepresented`
`  (pi : NonUnitalCStarRepresentation A H) : Prop`
`:=`
`  Function.Injective pi ∧ ∀ a : A,`
`IsCompactOperator (pi a)`
	For whichever algebra is represented, injectivity is faithfulness and the second conjunct is compactness of every represented element. Surjectivity onto all of $\mathcal{K}(H)$ is not required.

Existence of a star-isomorphism `Nonempty (A ≃⋆ₐ[ℂ] B) ↔`	`Nonempty` asserts that the type of $*$-algebra isomorphisms $A \to B$ has an element. The equivalence relates this existence to the existence in the next row; it selects neither a unique isomorphism nor a prescribed one.

Existence of a closure-space homeomorphism
`Nonempty (ClosureSpace.Homeomorph`
`(NormedSpace.geometricStructureClosure ℂ A)`
`  (NormedSpace.geometricStructureClosure ℂ B)) :=`
`by`
	The two full arguments are the geometric closures c_A on $S(A)$ and c_B on $S(B)$. `Nonempty` requires one homeomorphism $e : S(A) \to S(B)$ between these closure spaces.

The same map and inverse
`structure Homeomorph (c : ClosureSpace α) (d :`
`ClosureSpace β) extends α ≃ β where`
`  continuous_toFun : c.Continuous d toFun`
`  continuous_invFun : d.Continuous c invFun`
	The inherited equivalence supplies the bijection e and its inverse e^{-1}. The two fields require, for every $E \subseteq S(A)$ and $F \subseteq S(B)$ respectively,
	$$e(c_A(E)) \subseteq c_B(e(E)),$$
	$$e^{-1}(c_B(F)) \subseteq c_A(e^{-1}(F)).$$
	Both use the same e; together they give the closure equality in the Statement.

Exact surrounding source context (separate excerpts)

Exact surrounding source:

```
universe u v w

variable {A : Type u} {B : Type v}
variable [NonUnitalCStarAlgebra A] [NonUnitalCStarAlgebra B]
variable {H : Type w} [NormedAddCommGroup H] [InnerProductSpace ℂ H] [CompleteSpace H]
```

The two existential Hilbert spaces have the fixed size levels of their respective algebras (u for A, v for B). The independently sized space in the cited fixed-action theorem does not change these quantifiers.

Related exact source for the reading guide

Existence of a closure-space homeomorphism · Fixed source

```
      Nonempty (ClosureSpace.Homeomorph (NormedSpace.geometricStructureClosure ℂ A)
        (NormedSpace.geometricStructureClosure ℂ B)) := by
```

Meaning of compact representability · Fixed source

```
def NonUnitalCStarRepresentation.IsCompactlyRepresented
    (pi : NonUnitalCStarRepresentation A H) : Prop :=
  Function.Injective pi ∧ ∀ a : A, IsCompactOperator (pi a)
```

The same map and inverse · Fixed source

```
structure Homeomorph (c : ClosureSpace α) (d : ClosureSpace β) extends α ≃ β where
  continuous_toFun : c.Continuous d toFun
  continuous_invFun : d.Continuous c invFun
```